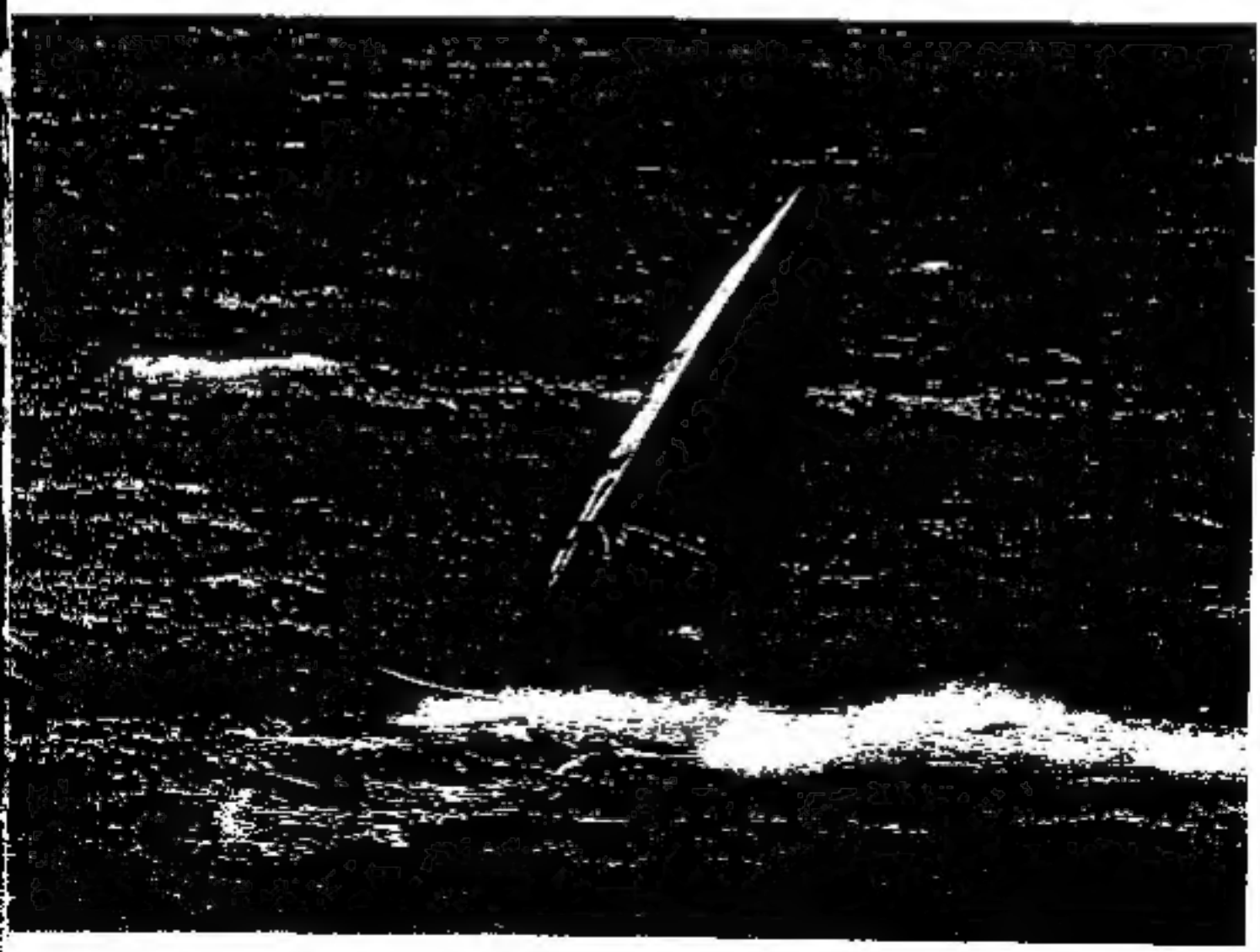
QLAVE

VOLUMEN V № 3 MAYO/JUNIO 1988



INFORMACION DEL CLUB

La integración en la asociación Qlave se hace por suscripción **ANUAL**.

El CEIUQL consta en el Registro Nacional de Asociaciones con el número 65210, y en el Registro Provincial de Zaragoza con el número 1742.

Qlave publica regularmente el boletín de los socios. Más información sobre la asociación puede obtenerse desde la **SECRETARIA** del Club.

Para ser miembro de Qlave se requiere estar interesado por el ordenador personal **SINCLAIR QL**.

El Club mantiene una **LIBRERIA DE SOFTWARE**.

La correspondencia debe enviarse al apartado de correos número 403/50080 de Zaragoza. Especificando si va dirigida a **SECRETARIA, TESORERIA, LIBRERIA** o si es una colaboración para publicarse en el Boletín.

GRUPOS LOCALES

ALBACETE	1	Rafael A. Bermudez (tlfno. 22 30 42).
ALICANTE	1	Blas Ortuño (tlfno. Zarandieta 7,4B).
GALICIA	1	Dario Carballeira (tlfno. 981 56 74).
MADRID	1	Enrique Hernandez (tlfno. 8 03 61 99).
MALAGA	1	Salvador Merino (tlfno. 47 30 48).
MALLORCA	1	Pedro Egea (tlfno. 41 82 38).
NAVARRA	1	Jaime Lacasa (tlfno. 23 69 86).
SEVILLA	1	Rafael Candau (tlfno. 12 02 64) 12274
		Jose M. Guzman (tlfno. 65 44 73).
VALENCIA	1	Enrique Sanchis (tlfno. 3 64 20 18).
VALLADOLID	1	Ignacio Enrique Cabero (tlfno. 33 59 82).
ZARAGOZA	1	Juan Alvarez (tlfno. 51 71 31).
		Jose Luis Forniés (tlfno. 35 54 65).

CONTRIBUCIONES AL BOLETIN QLAVE

Las contribuciones a Qlave deben ser ficheros QUILL en cartuchos de microdrives o en floppies de 3.5 preferiblemente. Las colaboraciones se devolverán a vuelta de correo. El número de líneas máximo por página es de 45 y el margen será a 90 columnas. Qlave no se hace responsable del contenido de los artículos firmados por su autor.

SE PROHIBE LA REPRODUCCION TOTAL O PARCIAL DEL CONTENIDO DEL BOLETIN

EDITORIAL

Hola, sufridos socios:

Se que estais esperando el Boletín impacientemente, pero entre que los que formamos la redaccion del Club, hemos estado de exámenes y ya sabeis que no nos sobra mucho tiempo.

El Club está siendo objeto de un gran interes por una oleada de usuarios del QL, que se enteraron por la revista QL WORLD de nuestra existencia, incluso usuarios de otros paises.

Desde aqui queremos agradecer a todos nuestro colaboradores, por su magnifica dedicacion al QL, y por sus mágnificos articulos. Estos genios que nos maravillan con sus conocimientos y su plena dedicación, son la sangre del Club. Nosotros somos el medio por donde se canalizan todos sus esfuerzos. Creemos que deberiais dar vuestras opiniones o realizarles preguntas sobre las posibles dudas que pudierais tener, ese motivarle aun mas sus esfuerzos y ganas de realizar más articulos.

Queremos desde aqui dar la bienvenida a los nuevos socios, y decirles que tienen el Club a su disposicion y que se les ayudará en todo lo que este en nuestras manos. Y que tomen ejemplo de nuestros Colaboradores, la condición de socio tiene unos derechos, pero tambien implica unos deberes, y si queremos que esté Club siga donde este, debemos todos incentivarlo y motivarlo para bién de todos nosotros.

Somos el Club de informática con más solera de España, y un gran Club; no todo es agua de rosas, sabemos que se podria mejorar, pero para ello necesitamos la colaboracion de todos y no de intereses particulares, aqui el único interés es el de todos, el de estar al tanto de todas las novedades y programas que realizán por el QL, y dar la mayor información posible a nuestros socios.

Diego Alcala
Presidente de QLave.

SALUDOS FERMOSOS
FELICES VACACIONES

CORREO DE LOS SOCIOS

Hoy 23 de mayo de 1988, despues de tres meses justos de haber recogido mi monitor de MISSA la ultima vez, mi monitor color BMC 1010DL se ha vuelto a averiar con la misma averia de siempre. Como la garantia ha caducado hace un mes, he decidido comprar un monitor monocromo Hantarex o Philips, pero aun no se cómo (voy a intentar comprarlo por mediación del corte ingles).

He tomado esta decision, porque de los dos años que hace que compre el monitor, cerca de la mitad de ese tiempo se lo ha pegado en MISSA (casi 11 meses).

Existen otros tres socios de Oiave con el mismo problema que yo. No voy a decir nombres, pero ellos saben a quienes me refiero. Creo que no hay mas socios con el mismo problema, porque no usan ese monitor, que creo que se ha ganado a pulso el calificativo de MALISIMO o ESTAFA.

Me he enterado demasiado tarde que Investronica está intentando liquidar el stock que le queda. Si alguien quiere tirar su dinero y comprar un monitor color sin garantias de reparacion que a los tres meses estara averiado, no se lo piensen mucho, compralo (y luego si lo ve conveniente, puede suicidarse).

Si desea comprar un monitor color le aconsejo una marca conocida, Philips.

Salvador Merino
Fuengirola (Oiave-154)

EN BUSCA DE LOS DOCUMENTOS PERDIDOS

Hoy 29 de febrero de 1988, me he enterado de que una gran cantidad de colaboraciones mías se perdieron en un desastre de dimensiones espectaculares (famoso corte de luz, o también metedura de pata con la Oram. Me confieso de haber metido la pata con la Oram formateando un disco en contra de mi voluntad). En realidad había recibido alguna insinuación de Serafín, pero no sabía a que se refería hasta que he recibido hoy el número de Diciembre 87 (enviado la semana anterior, porque por alguna extraña razon el otro se lo ha tragado la tierra). Menos mal que soy amigo de Indiana, y juntos hemos encontrado el MDV correcto en mi selva particular (solamente 152 MDVs).

Muchos socios se han quejado de que algún artículo era una traducción literal. Pido perdón por las molestias ocasionadas, pero era el precio a pagar por la rapidez (me he tomado la libertad de rectificar los siguientes artículos y eliminar algunas cosas poco interesantes). Esos artículos fueron escritos antes de perder mi equipo en Mayo 87 (lo he recuperado entero hace unos días), y las traducciones habían sido escritas en sucio directamente al ordenador. También me he enterado que algunos socios son lo suficiente inteligentes para darse cuenta que se habían perdido algunos capítulos del Forth.

Serafín dice que el S.L. de Zaragoza solamente se dedica a copiotes indiscriminado de programas comerciales. Me confieso de tener esa sana costumbre heredada del Spectrum (si ese ordenador no hubiese tenido la posibilidad de conseguir copias fáciles, habría sido un ordenador muerto en 1984) de coleccionismo con la idea de batir record (tengo 70 discos de 1.440 sectores llenos y unas 5.000 páginas. Creo que con un poco de dificultad podré llegar a final de año a las 10.000 páginas y 200 discos), pero también le estoy dedicando mi tiempo al QL con la idea de crear Software de dominio público. Yo creo que el coleccionismo, tal como existe dentro del Club, fomenta la compra de Software comercial nuevo y las relaciones entre los socios, pues es absolutamente necesario comprar nuevos títulos para poder seguir intercambiando (la mayoría de las veces se compran programas que no vamos a usar nunca, pero es la ley del consumo).

Por si alguien interpreta mal mis palabras, tengo que decir que soy enemigo número uno de esas sinvergüenzas que llenan sus bolsillos vendiendo programas que pertenecen a otras personas. P.e.: Valente. Antes de ser socio del Club le compré algunos programas creyendo que eran originales, luego no he vuelto a comprarle ninguno más.

Ahora los compro directamente a la casa distribuidora en UK. Pero creo que si varios amigos se ponen de acuerdo para comprar varios programas a pagar entre todos para uso colectivo es totalmente legal.

A continuación un volcado a la bestia de todo lo que los socios se perdieron hace mucho tiempo, pero esta vez será diferente.

Hoy 23 de febrero he conseguido recuperar mi monitor BMC Color. Llevaba tanto tiempo sin él que ya no me acordaba de su calidad de imagen. He habia acostumbrado a usar mi TV color portátil. En esa TV, podia visualizar 80 caracteres sin problemas (se podia usar los programas de Psion en modo 4), pero no podia visualizar los 85 caracteres, y los programas que usan esa caracteristica del QL son dificiles de usar con una TV.

Sobre HISSA, solamente puedo decir que son demasiado lentos. Un usuario oficial no puede estar privado de su equipo tantos meses.

Las noticias de Málaga son increibles. Aunque no he contactado con ninguno de ellos, es casi seguro que existe una especie de Club de amigos del QL en las universidades de Málaga (y bastante numeroso). Y yo durante los últimos años siempre habia creido en la posibilidad de ser usuario único en toda la provincia de Málaga.

He recibido de Quanta el FORTH 79, y resulta que es del mismo autor que el Superforth. El Forth 79 es una versión reducida del Superforth (Forth 83), pero que tiene un precio estupendo (2 libras + disco/MDV). Me han informado que existe más interés por aprender Forth que el que estaba pensando desde el primer momento. Creo que ya es hora de decir alguna direccion de interés, y voy a nombrar dos :

Forth Interest Group (FIGUK)
Coling Wallis
54 Wild Briar
Wokingham, Berks RG11 4UL

Revista Forthwrite

Forth Interest Group (FIG)
P.O. Box 8231
San Jose, CA 95155

Revista Forth Dimensions

FIGUK es más barato e interesante, economicamente hablando, que FIG.

Hoy es 25 de febrero, y aunque estaba avisado, he recibido por sorpresa la visita de Nacho Cabero (Valladolid). La primera impresion ya os la podeis imaginar (Nacho mide 2 metros o más de altura frente a mis 1.70 m). Es un chico bastante simpático y estuvimos hablando durante cerca de una hora del QL. Con él venia su hermana, y no sé que va a pensar de mi, pues nos olvidamos en nuestra conversación de su presencia. Ella trabaja en Marbella y sus abuelos estaban viviendo en los Bolsches (mi puente dormitorio). Nosotros hemos acordado seguir con nuestras relaciones por carta e intercambiar todo lo haga falta ( ) Oje ! aunque nos dediquemos a actividades heredadas del Spectrum, vamos a hacer cositas utiles.).

Salvador Merino
Fuengirola (QLeve-154)

FORTH-83 STANDARD (II)

VOCABULARIOS

FORTH (___) Vocabulario principal, contenido en todos los otros vocabularios; cuando ejecutado, será el primer vocabulario origen hasta que un usuario defina uno nuevo.

DEFINITIONS (___) La compilación vocabulario es cambiada para será la misma que el vocabulario el cual ha sido encontrado primero.

' nombre (___ad) Es usado en la forma ' <nombre> para buscar el diccionario por <nombre>. Si <nombre> es encontrado, entonces ad es la dirección de compilación de <nombre>. Por ejemplo, la dirección en la cual ha sido compilada en el diccionario cuando <nombre> ocurre en una definición colon. Si <nombre> no es encontrado un mensaje error es imprimido.

(') nombre (___) Es usado en modo compilación solamente, es usado en la forma (') <nombre> para buscar en el diccionario por <nombre>. Si <nombre> es encontrado, entonces la compilación de <nombre> es compilada en el diccionario como un literal. Por ejemplo, cuando más tarde ejecutada, esta dirección de compilación es izquierda en el stack.

FIND (ad1___ad2 n) Mira el contador de cadena guardado en ad1. Si no encontra
retorna 0, si la palabra es no-inmediata retorna -1, y si la palabra es inmediata retorna 1.

DEFINIENDO PALABRAS

CREATE (___) Sirve para definir palabras. Es usada en la forma CREATE <nombre> para crear una entrada en el diccionario llamada <nombre>. Cuando <nombre> es más tarde ejecutado, la dirección del parámetro del campo <nombre> es izquierda en el stack. Ejemplo: Para CREATE un diccionario entrada llamada FRED y para asignarle 6 bytes, escribimos CREATE FRED 6 ALLOT
O, si queremos guardar un mensaje en la forma de una cadena contada, CREATE MESSAGE 5 C, 72 C, 101 C, 108 C, 108 C, 111 C,
Para ver este mensaje, escribe MESSAGE COUNT TYPE
CREATE es usada por otras palabras, que sirven para definir otras, para crear entradas en el diccionario. Por ejemplo, la definición de variable es:
: VARIABLE CREATE 2 ALLOT ;
Una version alternativa, la cual no inicializa la variable a cero, es:
: VARIABLE CREATE 2 ALLOT ;

VOCABULARY (___) Es una palabra que define otras usada en la forma VOCABULARY <nombre> para definir un nuevo vocabulario, el cual cuando ejecutado, hará <nombre> el primer vocabulario a ser buscado cuando se este interpretando o compilando palabras.

CADENAS

CMOVE (ad1 ad2 un ___) Mueve un bytes de memoria desde la direccion ad1 a la direccion ad2, moviendo el byte en ad1 a ad2 primero, entonces procediendo hacia mas alta memoria.

CMOVE) (ad1 ad2 un ___)Como CMOVE, excepto que el byte en la direccion (ad1+un-1) es movido a (ad2+un-1) primero, entonces procediendo hacia más baja memoria.

FILL (ad un n ___) un bytes de memoria, comenzando en la direccion ad hacia arriba, son ajustados al menos significante byte de n.

COUNT (ad ... ad+1 n) Deja el caracter contador n, guardado en la posicion memoria ad, como TOS e incrementa ad para dejar ad+1 como 205. Como puede ser visto, el stack esta ahora en el correcto estado para TYPE. La secuencia COUNT TYPE es comunmente usada.

-TRAILING (ad n1 ___ ad2 n2) ad y n1 son la direccion y caracter contador de una cadena de cadena. -TRAILING reduce el contador por el número de caracter espacio en el fin de la cadena para dejar un nuevo caracter contador n2. La cadena guardada en memoria sigue sin cambiar.

CONVERSION NUMERICA

BASE (___ ad) Es una variable que guarda la base actual para la conversión de números, ambos input y output.

DECIMAL (___) Carga decimal 10 en BASE

CONVERT (d1 ad ___ d2 ad2) La cadena de caracter empezando en ad+1 es convertida y acumulada en d1 para dar d2, convirtiendo la cadena en digitos, caracter por caracter hasta que un no convertible caracter es encontrado. Ha medida cada caracter es convertido, d1 es multiplicado por BASE y el digito sumado. ad2 es la direccion del primer no convertible caracter.

<# (___) Comienza numerica output conversion.

(d1 ___ d2) d1 es dividido por BASE y el resto convertido a un ASCII caracter el cual es entonces guardado en la dirección próxima más baja en PAD en el fin de la cadena que está siendo convertida. Ambos d1 y d2 deben ser positivos doble enteros.

#> (d ___ ad n) Deste d y deja la direccion y contador de la cadena, formada por usando # y/o #S en PAD. ad y n juntos son conforme por TYPE.

#S (d ___ 0 0) Convierte d a una cadena de ASCII caracteres guardados en PAD. Si d es 0, entonces un unico caracter 0 es adjuntado a la cadena.

HOLD (n___) Salva el menos significante byte de n en PAD como parte del output de la cadena que está siendo convertida. Tipicamente usada entre <# y #>.

SIGN (n___) Si n es negativo, un ASCII signo menos es sumado al comienzo de la cadena PAD, tipicamente usado entre <# y #> despues un número ha sido completamente convertido.

Por fin, he terminado de definirlas todas. (la verdad, no me gusta traducir y siempre la teoría. Solamente me gusta la práctica). SUPERFORTH tiene más de 400 palabras definidas en el diccionario base, pero los que usen el SUPERFORTH ya las tienen explicadas en el manual (más que un manual, parece un libro).

Si algún interesado tiene alguna duda, podemos intercambiar experiencias.

Si alguien ha visto en alguna librería un libro dedicado al Forth en Castellano (en inglés, existen más de 30), le ruego que me lo comunique.

Solamente me queda decir que ha comenzado la guerra. voy a enviar programas en Forth a la librería del Club, pero voy a necesitar ideas y alguna información.

Salvador Merino
Málaga (Olave-154).

PRACTICANDO CON EL SUPERFORTH (I)

Hace ya varios meses, me decidí en compartir con todos los socios de Olave mis conocimientos en Forth-83 (totalmente gratis, y por Hobby). Solamente he tenido respuesta afirmativa de 7 socios (eso era en junio 87. Ahora no tengo ni idea), muy pocos, pero suficientes si tenemos en cuenta el poco software que tienen la mayoría de los socios y la cantidad de libros en castellano publicados sobre Forth-83.

Me he tomado la libertad de poner al día la actualidad del mundo Forth, ya que en nuestro país, parece que se han olvidado de este lenguaje (quizás uno de los faciles), como con nuestro QL.

Una vez, os dije que la continuación sería por la vía práctica, pues si alguien duda de mi palabra, continue leyendo.

Cuando dividimos un número de 32 bit por uno de 16 bit, el resultado es un resto de 16 bit y un cociente de 16 bit. Pues si el número de 32 bit es dividido por uno de 8 bit, el resultado puede ser un resto de 8 bit y un cociente de 32 bit. En este último caso, la instrucción UM/MOD da como resultado un error 15 division overflow (en el Superforth). Para solucionar este problema tenemos que definir otra palabra, y siguiendo un ejercicio de principiantes de FIG, he definido la siguiente:

```
: UM/MMOD ( UD1 U2 - U3 UD4 ) DUP 65536. ROT UM/MOD 3 ROLL DUP ROT UM* 2SWAP UM* 5
ROLL 0-D D+ 4 ROLL UM/MOD 5->D 4 ROLL 4 ROLL D+ ;
```

Naturalmente visto así no se entiende (lógica es uno de los defectos del Forth), pero veamos el programa funcionando paso a paso (es muy importante conocer el estado del stack en todo momento, y saber que un número de 32 bit se almacena con dos entradas A B, siendo el número de 32 bit el resultado de A+(B*65536)).

P.e: 999424. 9 UM/MMOD , el resultado D. 139884 . 4

```
( 16384 15 5 ) DUP 65536.     ( 16384 15 5 5 0 1 ) ROT UM/MOD
( 16384 15 5 1 13107 ) 3 ROLL DUP  ( 16384 5 1 13107 15 15 ) ROT UM*
( 16384 5 1 15 65533 2 ) 2SWAP UM*  ( 16384 5 65533 2 15 0 ) 5 ROLL
( 5 65533 2 15 0 16384 ) S->D D+  ( 5 65533 2 16399 0 ) 4 ROLL UM/MOD
( 65533 2 4 3279 ) S->D  ( 65533 2 4 3279 0 ) 4 ROLL 4 ROLL D+ ( 4 3276 3)
```

Esta palabra ha sido donada a la librería del Club junto con la del siguiente artículo en el fichero MATHS_FTH (escrito con el editor del Workbench). Ese fichero está preparado para ser usado con el Superforth. Os puedo garantizar que ahí hay más (P.ej: tengo que probar unas palabras para crear matrices, para poder traducir unos programas en Basic de un amigo de Pamplona. Es una gran prueba de fuego, pero no imposible).

Salvador Merino
Málaga (Qlave-154)

PRACTICANDO CON EL SUPERFORTH (II)

Cuando multiplicamos un número de 32 bit por uno de 8 bit, nos encontramos con que no tenemos una palabra para hacer eso. Pues bien, he aquí una palabra que lo hace :

```
: UM*** ( UD1 U2 - UD3 ) DUP ROT * >R SWAP 2SWAP UM* D+ ;
```

Veamos un ejemplo del programa paso a paso:

```
983063. 5 UM***     da      D.  4915315

( 23 15 5 ) UM*** DUP ( 23 15 5 5 ) ROT ( 23 5 15 5 ) * >R SWAP
( 23 5 0 75 ) 2SWAP ( 0 75 23 5 ) UM* ( 0 75 115 0 ) D+
```

Esta palabra se puede usar, pero tiene sus limitaciones (p.e.: si el número UD1 es superior a 24 bit, puede ocurrir un overflow (depende del tamaño de U2) si al efectuarse la operación el resultado es superior a un número de 32 bit.

Solamente me queda decir que estas cosas no pasan en un sistema forth de 32 bit, porque el límite de cálculo es muy alto.

Salvador Merino
Málaga (Qlave-154).

MAS SOBRE LA TRUMP CARD

Usando el FREE_MEM del Toolkit I, la memoria libre es 857.088 bytes. Y en modo 128 K la memoria libre es 84.480 bytes.

La única diferencia entre modo 896 K y 128 K es que en modo 128 K tenemos 768 K menos en el mensaje de presentación se cambia 896 por 128). Todo lo demás está disponible en ambos modos a excepción de las utilidades extra en ROM de la GRAM

Se pueden crear un máximo de 8 Ram Disk, pero está limitado a lo memoria d spon ble se puede elegir el número de sectores) Otra utilidad es el Microdrive Imaging, probando he cargado el mdv QUILL entero en 10 segundos (no está mal), y después he cargado QUILL desde RAM Disk con emulador) en menos de 2 segundos, aunque para poder usar esas ventajas en QUILL hay que modificarlo un poco (el programa de Rafael Gandos puede servir , pero con la Qram no hay problema (la Qram es totalmente necesaria para sacarle el máximo partido a esta gran cantidad de memoria).

El QL se me ha convertido en una máquina muy grande, y con muchas cosas que probar y explorar. Es increible, tengo el QL en mi casa desde hace 21 meses (octubre 87) y aún no he probado ni el 20% de sus posibilidades

MAS SOBRE EL SUPERFORTH

No he podido probar el Superforth por causas conocidas por todos. Pero cuando lo he probado (segunda semana de septiembre), he tenido problemas (por cierto, bastante tontos que por una casualidad del destino (como siempre) las he podido solucionar. El Superforth a diferencia del Forth V 1.3 da un mensaje de error sin especificar donde exactamente ha ocurrido (es tan rápido detectando un error que lo cuesta detectar antes que ocurra , y en el último (Por cierto, un Forth-83 incompleto y mezclado con Forth-79 comunica en que linea y BLK se ha producido

El editor del Superforth está escrito completamente en Forth-83. Se invita al usuario a, lo desee a modificarlo

El paquete de Floating Point Maths está escrito completamente en Forth-83. Si las palabras no fuesen las mismas que en el standard de FIG, el usuario las podría cambiar fácilmente Hasta yo mismo estoy sorprendido, y creo que me queda mucho que aprender estoy en pañales aún).

El Superforth almacena los programas en BLK (si se usa el editor, y es la forma standard) o en ficheros. En este último método, el fichero se puede crear de varias maneras, que son las siguientes :

- Utilizando la utilidad del BLK4, que consiste en unir varios BLK en un solo fichero.
- Utilizando QUILL, pero tendrá que instalar una impresora con el INSTALL_BAS

s gu ando as instrucciones del manual.

-Utilizando un editor de los buenos. A mi, me gusta el editor del assembler Workbench por su rapidez, y porque puedo lanzar los dos programas a la vez (me sobra menos a por todos lados)

El ejemplo para introducir codigo máquina está escrito con el ensamblador de Metacomco pero podemos utilizar todos los ensambladores disponibles (en mi máquina me caben a elegir

NOTICIAS DEL MUNDO FORTH

La proxima Formal Conference sera en ASILOMAR CONFERENCE CENTER, MONTEREY PENINSULA OVERLOOK NO THE PACIFIC OCEAN, PACIFIC GROVE, CALIFORNIA. Los dias 27-29 de noviembre de 1987. El gran tema: Forth y los ordenadores de 32 bit. Los subtemas son:

-Espacio grande de direccion en ordenadores de 32 bit
-Display gráfico, windows, y manejo de menus.
-Relación de sistemas operativos, otros lenguajes, y networks
-Control de estructuras, data estructures,objetos, y cadenas.
-Ficheros, gráficos, y operaciones en coma flotante.
-Comparaciones con ordenadores de 16-bit.

Para más informacion habrá que esperar a 1988. Para la pregunta , ¿ Usan OLs ?. La respuesta, casi desconocido (pero existen algunos usuarios, por ejemplo yo mismo). Para la transición a máquinas de 32-bit ya un hecho. Este lio se ha formado al aumentar el numero de socios con máquinas de 32-bit.

El lenguaje Forth está en manos de los hackers, que aprecian mucho mas este lenguaje por su rapidez (es mas rapido que el C y el Pascal) y control total de la máquina, y la industria.

Los usuarios del Forth usan maquinas de todo tipo, siendo increible la cantidad de ordenadores diferentes y sistemas Forth que existen en una comunidad de usuarios tan reducida (algunos se han construido su propio ordenador, y se han hecho su propio sistema. Es increible

NOTICIAS DEL MUNDO FORTH

Hace ya mucho tiempo, para ser exactos 1984, unos socios Españoles de FIG publicaron algunos cursillos de Forth-79, y anunciaron que se había hecho el Forth-83 Standard. Ahora, sin saber nada sobre el paradero de esos socios, se anuncio que el Forth de 32 bit existe desde hace un año.

Según 1987 ROCHESTER Forth Conference, hace ya bastante tiempo que existen los sistemas Forth de 32 bit, y este año ha aumentado considerablemente el número de usuarios de máquinas de 32 bits.

Esos usuarios están totalmente convencidos que los dias de los 16-bit ers están

contados las posibilidades, que se han abierto con el salto cuántico en velocidad de los nuevos Forth processors, son increibles, y ya las he comentado en Forth en el Futuro. Tengo una lista de todas las nuevas palabras que manejan el stack de 32 bit con sus equivalentes en Forth-83 totalmente definidas y explicadas. Si hay algo interesante en un sistema Forth de 32 bit es su comodidad y su potencia de calculo.

El QL y su familia podría usar un sistema Forth de 32 bit, porque internamente maneja números de 32 bit y gracias a su arquitectura puede usar sistemas de 8 y 16 también.

El OS/2 o LA ESPERANZA DE LOS PCs

El MS OS/2 lo está vendiendo Microsoft Corp en forma de Toolkit. Naturalmente, ese Toolkit es una versión sin terminar de la versión completa, la cual no se ha terminado aún. El MS OS/2 es un sistema multitarea monousuario, por lo tanto el QDOS y el $,M % no tienen nada que envidiarle. Pero existe una diferencia, estos dos últimos están terminados y disponibles. Durante mucho tiempo, los usuarios de QL fuimos sido los únicos usuarios de ordenador personal económico que hemos podido correr varios programas en multitarea, y usar impresora y disco a la vez. En otros ordenadores (siempre y cuando no usen UNIX, que es más caro que un QL), un programa cuando usa la impresora no puede hacer otra cosa hasta que termine. En un QL se puede estar imprimiendo un documento en la impresora y pasando ficheros de un disco a Ramdisk a la vez (y escribiendo en otra QUILL, incluso).

Microsoft ha anunciado el libro Advanced OS/2 Programming para enero 1988 (año en el que se piensa terminar el sistema operativo). Yo creo que el Futura puede aún cambiar el mercado (aunque ahora tengo más fe en el CST THOR XVI). Primero el Toolkit OS/2 se vende a 3.000 dólares (más del doble que un Futura Básico), y segundo la posibilidad de ejecutar muchas aplicaciones MS-DOS existentes, pero sin beneficiarse de memoria adicional. Esto último lo puede hacer el Futura con una adaptación del IBM PC emulador disponible para los 68000 (tienen una versión los Atari ST, Apple,...)

Yo creo que el Futura se va a vender muy bien al principio (¡Dios lo salve de la competencia de los otros 68000!) y puede ser un competidor muy poderoso con máquinas que presumen de grandes. Eso sí, la crítica será muy dura, y se intentará por todos los medios hundirlo. Pero existen muchos Fans del QL que nunca abandonarán a ninguno de los dos.

Al estar todos estos artículos escritos antes de Octubre 87, las noticias han cambiado mucho desde aquellos lejanos días. Por ejemplo, el Futura tiene tantas posibilidades de salir al mercado como yo de aprender CHIMO. Pero no se preocupen, el CST THOR XVI incorpora la mayoría de las posibilidades del Futura. Por lo tanto, ¡LARGA VIDA PARA EL QDOS!.

Salvador Merino
Málaga (Qlave-194)

MAS SOBRE RESPR

Cuando recibí el número de Diciembre 87, uno de los artículos que más me llamó la atención fue « COMO JUGAR AL AJEDREZ MIENTRAS SE ESCRIBE SOBRE CODIGO RELOCAL ZABLE » firmado por Ernesto de Jesús Alcañiz, que es amigo mío por correo! Una vez leído me puse a meterle mano a mi versión del CHESS, y los resultados son los siguientes

Mi QL está ampliado con la TRUMP CARD y por lo tanto muchas cosas escritas en ese artículo no valen en él. Mi QL en modo 896K tiene libre antes de hacer un D_R FLP1_ 857 088 bytes libres y después 855.552 bytes libres (primera dirección disponible para el usuario es la 153.024) . En modo 128K tiene antes de hacer un D_R FLP1_ 84 480 bytes libres y después 82.944 bytes libres (primera dirección disponible para el usuario es la 129.200).

En primer lugar, tengo que decir que no he encontrado aún un programa que no corra con la Trump Card (puede correr en modo 128K o 896K). En el caso de JROOM se ignora el BOOT y se pasa directamente al EXEC_W (valiente rollo se ha marcado el programador con los RESPR).

En mi QL, en modo 896K un RESPR(0) equivale a 1048576-0=1 048 576 (1024 K) En modo 128K un RESPR(0) equivale a 262144-0=262.144 (256 K). El RESPR funciona restando la cantidad entre paréntesis a la posición de memoria más alta disponible y el resultado es la posición donde se va a cargar el programa. En el caso del CHESS después de los dos RESPR nos sale una dirección reservada al sistema (nada menos que a 184.064 .

Mi versión del CHESS desprotegida con la rutina de la librería funciona estupendamente en modo 128K, pero no funciona en otra dirección (pues aunque aparece el mensaje de desprotegido, el programa salta al mensaje de error de original no encontrado) seguramente porque al hacer no tuve en cuenta que el código era relocalizable (creo que el autor modificaba la desprotección para que corra en cualquier dirección) La versión protegida funciona sin problemas en modo 896K en otra dirección.

El programa JOIN de la librería lo único que hace es un simple LBYTES FDK1_CHESSC, 184064 (pero existe un pequeño defecto) Los resultados son los mismos en los dos casos, y solamente se puede usar en modo 128K El programa funciona sin problemas en modo dos dimensiones, pero si ponemos a modo tres dimensiones se cuelga esto es válido para la versión protegida y desprotegida)

Salvador Merino
Fuengirola (QLave-15# .

BUEN TRATO POR CORREO

Hace unos cuantos meses encargué a la casa Miracle una expansión de memoria de 512 K. Recibí dicha expansión en un breve plazo de tiempo y el cobro se realizó a través de Visa sin ningún problema. Sin embargo he tenido algunos problemas de calentamiento del QL al tener conectada la expansión de memoria. Decidí buscar ayuda en el propio fabricante que de seguro debía tener información al respecto; mandé una carta a Miracle explicando mis dificultades y todavía estoy esperando la contestación desde el mes de Febrero, procurando buscar soluciones por mi propia cuenta.

La historia a que os acabo de contar me ha planteado ciertas dudas a la hora de volver a realizar compras por correo; a pesar de todo volví a decidirme hace algo más de un mes. Encargué a QJump su QFLP para mi interface Microperipherals, con la idea de comprar más adelante su QRAM y animado por disponer de algunas de las extensiones del Super Toolkit II que se incluyen en el QFLP.

Recibí rap días mamente el chip y me dispuse a probarlo ¡Cual no sería mi desesperación al comprobar que no consegula hacer funcionar los comandos de selección de opciones de nivel de seguridad, de tiempo de arranque y de número de pistas, ni tampoco ninguna de las extensiones al Superbasic!

Sin muchas esperanzas me dirigí por carta a QJump exponiendo mis desgracias. Mi sorpresa fue cuando al cabo de pocos días recibí una carta dándome todo tipo de explicaciones y firmada personalmente por Tony Tebby.

A continuación os traduzco la carta para que pueda servir de ayuda a quien tenga los mismos problemas y para mostrar públicamente mi agradecimiento por su practicidad y cortesía al Sr. Tebby

" Estimado Mr de Prado

Acabamos de descubrir dos errores graves en las notas que acompañan al QFLP por su interface de disco de Microperipherals El primero es que la ampliación tiene un único comando para seleccionar las opciones del floppy en vez de los tres comandos separados FLP_SEC, FLP_START y FLP_TRACK. Este comando tiene tres formatos:

FLP_OPT nivel de seguridad
FLP_OPT nivel de seguridad, tiempo de arranque
FLP_OPT nivel de seguridad, tiempo de arranque, número de pistas

El segundo error es que el comando que activa las extensiones al SuperBASIC (FOPEN, FLEN etc ha sido omitido. Este es FLP_EXT Si coloca el comando FLP_EXT en un archivo boot para ROMs AH ó JM el comando FLP_EXT puede no estar en el mismo archivo que cualquier declaración en SuperBASIC que use una de las extensiones.

Debo pedir excusas por la confusión causada por estos errores

Sinceramente suyo

Tony Tebby "

José Carlos de Prado
Madrid Qlave (216)

En primer lugar un saludo afectuoso a todos los socios de este club en especial a aquellos que con su esfuerzo hacen posible su funcionamiento y la publicacion de Spin-n

Perdonenme mi pobre estilo literario pero la posibilidad de comunicarme con un gran colectivo que comparte conmigo la aficion a nuestro ordenador,el poder preguntar una y mil dudas que siempre me tenido,el contar las experiencias sufridas,los errores cometidos e incluso hace que me decida a escribir esta nota con el ruego de su publicacion.

Debido al aislamiento que he soportado en dos años desde que poseo el QL no deseo ponerme en contacto con todos los socios,pero como se que esto es imposible por razones practicas,ruego que publiqueis mi direccion y num.de telefono para que aquellos socios que de alguna manera comparten mis aficiones puedan escribirme.

Quiero aclarar que mi formacion informatica es bastante precaria,autodidacta y con las lagunas propias de aquel que sin una base se decide a releer todos los libros que sobre el tema encuentra y que la mayoria de las veces apenas las entiende correctamente. Tuve la suerte de a traves de una conocida revista de Spectrum ponerme en contacto con un socio de este Club que reside en Jerez de la Frontera Para mi fue un encuentro de enriquecimiento por lo que me supuso de ayuda.

Como solo tenia un conocido con este mismo ordenador,dudaba de que en mi circulo de colegas Muchas gracias Angel por tu paciencia conmigo!. A continuacion os describo mi equipo que pongo a vuestra disposicion Como ordenador tengo la version MGE del QL ampliacion de memoria de 512 k,monitor Zenith de fosforo verde y TV SUPER/SHARP 14" mod.f,cedido como monitor color,estabilizador de tension e impresora termica de 40 columnas y con interface serie v21.

Tambien dispongo de una doble unidad exterior de MOVES procedente del Spectrum y que con un cable cinta largo (30 cm.) manejo como Mdv2_ y Mdv4_. Ahora estoy en la espera de recibir una unidad de disco 3.5" que completara mi equipo.

Como fui incapaz de conseguir en Murcia conectores machos del tipo que equipa la maquina,al fin,decidi ponerle en paralelo con las actuales otros de tipo CANON "D" de 9 pines,los de joystick los cables tipo Atari para aprovechar los que tenia del Spectrum Las dos salidas serie los cables como la salida serie del INTERFACE1 del Spectrum tambien con lo que me facilito el poder unir ambos.

La consecuencia de esta modificacion fue que mediante un programita en basic y una pequeña rutina en CM.resolvi el grave problema de tener copias de seguridad de mis programas de QL a la utilizar MOVES ya que las guardo en cintas de cassette y me caben 6 Mdvs en una cinta C-60

Ahora cuando disponga de los discos ya no me sera interesante esta operacion pero ahi queda como experiencia. También tengo realizadas algunas experiencias telemáticas entre dos telefonos mediante modem como via radio hay radioaficionados también. Por tener disponible dos MODEM full duplex 300 baudios con respuesta automatica y acceso directo para probar entre ellos que dispongo de un programa de comunicaciones del que carezco quisiera realizar cualquier tipo de experiencias en este sentido.

Y ahora un favor! Por accidente se me borraron parte de las instrucciones en ensamblador del compilador de C de DSI quedando inutilizadas ¿Alguien podria suministrarmelas? Gracias

Una pregunta Al no haber en Murcia ningun Grupo Local, pese a yo disponer de abrir a de programas a traves de las de Alicante o Albacete que no son mas pero mas?
Otra mas: Esta todavia vigente la oferta de Investronica para la compra del raton y podre apuntarme para el proximo lote?

Si a alguien le interesase alguna de estas modificaciones tanto en la maquina como en la TV que no dude solicitarmelas por escrito o via boletin

Un saludo afectuoso

Antonio Rodriguez Hernandez(Olivo-84)
Apartado 2147 Tfno 968-802157
30080 MURCIA

Cuando recibo estas líneas aún no he recibido el boletín nº 2 de este año. Espero que el anuncio de que se convertía en bimensual no pase a trimestral o cuatrimestral. En todo caso y para poner mi granito de arena os envío otra colaboración sobre lenguaje Forth sigo dándole vueltas a la cabeza en torno a cómo aumentar la actividad y participación de todos los miembros.

En este último sentido os propongo abrir una nueva sección en el boletín. Se trataría de una sección de CONTACTOS. No, no me entendáis mal, no me refiero a esos contactos en que estáis pensando, se pretende darle un tinte erótico a QLave, en realidad yo propondría reservar una página para que los socios, mediante unas pocas líneas, pudieran anunciar sus deseos de contactar con personas concretas dentro del club o con un determinado sector o grupo o de recibir determinadas informaciones de algún otro socio.

Pienso que este tipo de anuncio requiere un esfuerzo mínimo por parte de quien lo contesta y a cambio puede facilitar la solución a muchos de los problemas con que uno tropieza al tener tratos con estas máquinas, especialmente si uno se encuentra solo en la tarea.

Para predicar con el ejemplo ahí van mis dos primeros anuncios para la sección de CONTACTOS:

Loco por la programación en distintos lenguajes desea ponerse en contacto con otros que compartan la misma locura. Estoy especialmente interesado por el Forth y por el lenguaje C. Me han llamado mucho la atención las colaboraciones de Salvador Merino de Málaga.

Desearía ponerme en comunicación con otros socios de Madrid. Lo he intentado hace ya dos meses por el teléfono que figura en el boletín y aún espero la respuesta.

Mis señas son :

José Carlos de Prada
C/ Hacienda nº 16, 3º D
28019 MADRID

Teléfono: 4-69-96-43

José Carlos de Prada
Madrid (QLave 216)

Acabo de recibir la revista de Marzo-Abril y me siento en la obligación de felicitar a todos por el trabajo realizado. Excelente el curso de Superbasic, erratas aparte. Ahí es donde se colaboradores de Salvador Merino, de Nacho Cabero y de todos los demás.

Como socio pasivo que soy sólo una colaboración en años y medio, viendo arrastrando un sentimiento de culpabilidad por la progresiva decadencia en que se veía sumergido nuestro club. Me he dicho muchas veces. Pero qué tengo yo que contar referente al QL que me haga pensar publicando después de leer EL FINAL DEL STRIP POKER de Salvador Merino con su buena dosis de humor o ese que ninguno de nosotros se puede hacer esa pregunta.

Todos los que alguna vez nos hemos puesto delante de nuestro ordenador tenemos cosas que decir. Trucos, ideas, comentarios de programas, preguntas, porque no todos son expertos en informática pero seguro que hay por ahí miles de dudas y preguntas que nos hacemos referente a nuestro ordenador y las preguntas generan respuestas y éstas generan sobretodo COMUNICACIÓN que pienso que es de lo que más adolece nuestro CLAVE.

Pequeña a poco vamos mirando, la entrega de tanta buena gente que 'trabaja' en nuestro club, por citar un caso reciente ahí está Serafín Olcoz. La verdad es que es triste que se 'quemen' por la desidia y el desinterés de nosotros, los socios pasivos. Aquí me viene a la mente la frase con que se despidió Serafín la última vez que hablé con él por teléfono: AUPA EL QL.

Me, pues va ahora por vosotros. ¡AUPA A TODOS LOS QUE TRABAJAIS POR CLAVE.

Víctor Librero
(CLAVE-184)

GRACIAS FERNOSO

PREGUNTAS DE LOS SOCIOS

Tengo un pequeño problema con mi Trump Card y espero que algún usuario de ella me cuente si le pasa lo mismo o sabe algo del tema: al cargar el Abacus, antes de completarse la pantalla de trabajo me llegan a aparecer las opciones de la parte superior, se marcha en la zona inferior el indicador de celda y de memoria, bye 640 Kbytes y rápidamente se borra para dar paso al error típico del paquete de Psion con el mensaje '32' y 'pulse ESPACIADOR para continuar'. Como es de esperar, si se pulsa SPACE vuelve a repetirse el proceso

Solo he conseguido hacer funcionar el programa incluyendo un REPSR 684000 en el boot otro valor menor de 684000, más o menos, no funciona), con lo cual luego quedan libres como 90 Kbytes más de los 15 que quedan con 128 Kbytes pero menos de los casi 700 que deberían ser con la Trump Card. También funciona sin problemas si hacemos un RES_128 para anular la ampliación de memoria de la Trump Card.

Por supuesto, lo mismo ocurre si se carga el Abacus desde microdrive. Esto no me ocurre con el Quill ni el Easel ni el Archive, pues todos trabajan a la perfección con la ampliación de memoria y la unidad de disco. ¿Alguien sabe algo de esto? No es que haga mucha uso del Abacus, pero me gustaría seleccionarlo

También me gustaría que quien emplease teclado diferente del original del QL mandase algunas líneas comentando las características del mismo (teclas desdobladas, teclas adicionales tacto ..) lo más detalladamente posible, así como si ha tenido problemas para adaptarlo Estoy interesado en algún buen teclado para el QL pero no me arriesgo a pedirlo sin conocerlo algo más de lo que es posible a través de la publicidad o artículos del QL World. Así que, animaros los que lo tengáis y contad vuestras peripecias sea el teclado que sea En concreto me preocupa el tema de los caracteres españoles, pues supongo que habrá que cambiar algunas teclas de sitio, si es posible, o poner más bien etiquetas sobre ellas

He escrito un ensamblador del Z-80 para el QL con el objeto de poder continuar escribiendo un programa bastante extenso que estoy realizando para el Spectrum. Lo he compilado con Turbo y lo ejecuto conjuntamente con The Editor, pero esto me da algunos problemas. Lo único es que no sé si como funciona, pues las extensiones que añade The Editor (del Supercharge) deberían crear conflicto con las del Turbo Toolkit

Al principio The Editor se detenía por 'desbordamiento' al intentar leer o grabar en FLP_ o RAM_ pero cargándolo en primer lugar y dejando pasar unos segundos antes de instalar el Toolkit II (que también es empleado en el ensamblador del Z-80, el Runtime Turbo Toolkit runtime_exts) y el ensamblador, parece que no crea problemas. Solo de vez en cuando se queda todo bloqueado, cuando en algún EDIT$ del ensamblador se responde con el cursor; no sé si es debido a lo mismo

Si alguien sabe algo de esto, le agradecería que me lo aclarase. Puede ser que The Editor sea un programa compilado con Turbo? ¿Alguien me puede aclarar qué diferencia hay entre instalar Turbo Toolkit o su versión runtime? Si alguien conoce el Text 87 u otro editor de textos "EXECutable" que no tenga que ver con Turbo, le ruego que me cuente si se puede trabajar con textos ASCII puros como con The Editor o si tiene tanta flexibilidad y rapidez como éste.

Por último, he oído que la serie de Forth corre el peligro de desaparecer. Desde aquí animo a Salvador Merino para que siga adelante. Soy seguidor del Forth desde hace tiempo. En concreto he trabajado con el Specforth de Artic y el Abersoft Forth de Melbourne. Fig-Forth para Spectrum y más recientemente he hecho algo en FORTH-83 con el de Lambda en el QL y seguramente pediré el Superforth de D.P.

Tengo algunos temas interesantes pero que me llevará algún tiempo ordenar y adaptar, que están en Fig-Forth para Spectrum, pero que enviaré bien documentados en cuanto los tenga preparados. Una pregunta directa, supongo que a Salvador Merino acerca del Forth Interest Group: cómo hay que hacer para entrar en contacto con ellos? Qué actividades hay por aquí sobre este tema?. Te ruego que me pongas al corriente de todo esto. Otra cosa: ese Forth de 32 bits del que has comentado algo, existe realmente ya para algún ordenador o es sólo una idea?

Marcos Cruz (OLave-244)
Madrid, Mayo 1988

He pedido varios programas de la librería y he observado que hay bastantes programas de calidad pero con algunos no me aclaro mucho a la hora de utilizarlos a pesar de los comentarios que aparecen al principio de los programas y a veces los ficheros DUMP que vienen en alguno de ellos. Propongo que en la sección: Comentario de programas se comenten dichos programas, a ser posible por los propios autores. Os aseguro que las peticiones de dichos programas aumentarían considerablemente.

Ahora una pregunta que me interesa desde hace tiempo y que el artículo ORDENAR UN BUSQUEDA DE DATOS publicado en el último número me ha reactivado ¿Es posible hacer un programa que genere GRAFICOS mediante una base de datos de valores? ¿Se podrían almacenar esas bases de datos, que me imagino que serían extensas, en nuestros queridos microdrives?

Para finalizar un comentario: A todos aquellos que se les estropeen la membrana del QL me dirá que HISSA MADRID la suministra a 1037 ptas (esto una por persona) mas gastos de envío así está el problema. Pues la envían por SEUR y puede llegar a costar casi la mitad más, como ya ha apuntado algún socio es facilísimo colocarla.

Victor Librero
(QLave-184)

TRUCOS

Lo primero de todo, decir que la idea del truco no es mia, sino de mi amigo Cesar Conde de Alcalá de Guadaira, y posible futuro socio de QLave (una version parecida de este genial truco fue publicado en QUANTA Volume 1 ISSUE 10 November 1994 en las páginas 26-27). El programa se basa en usar las rutinas de volcado de pantalla a impresora que vienen con el programa EASEL de Psion (la impresora usada es la EPSON serie compatible con IBM):

```
20 MODE 4
100 WINDOW 512,256,0,0:PAPER 0 CLS
105 c$=INKEY$
110 a=RESPR 2500:
120 INPUT   Tamaño de la volcada?(c/n) ';c$
130 CLS
140 LBYTES mdv1_nombre,131072
150 RECOL 7 6,5,4,3,2,1,0
160  IF c$='c' THEN LBYTES mdv1_grint_prt,a
170 IF c$='n' THEN LBYTES mdv1_fx100_prt,a
180 CALL a
```

Las líneas de arriba hacen lo siguiente:
- Creación de una window que ocupe toda la pantalla.
- Carga del mdv el dibujo que queremos volcar
- Recoloreamos el dibujo. Se le puede dar la entonación deseada.
- Se pide el tamaño del dibujo (comprimido-normal).
- Se carga del mdv el fichero de impresora elegido.
- Se ejecuta con Call.

El programa tarda 5-6 minutos en imprimir el dibujo, y los resultados son muy buenos. Más de uno se han gastado su dinero en programas comerciales o se han vuelto locos escribiendo una rutina en código máquina sin saber que tenían el trabajo hecho desde el dia de la compra de su QL.

Tengo que recordar que los usuarios de Trump Card poseen comandos de volcados en la ROM son los mismos que la QRAM, pero desde el Superbasic)

Salvador Merino
Fuengirola (QLave-154 .

COLABORACIONES DE LOS SOCIOS

PRACTICANDO CON EL SUPERFORTH (VII)

Este artículo se lo voy a dedicar a mi amigo Nacho Enrique Cabero Valladolid.

El tema de este artículo está basado en el código máquina. Si tuviese que ganarme la vida programando en assembler, casi seguro que me moriría de hambre, pues no soy un experto programador de código máquina.

En Forth cuando programamos en Código maquina, lo que hacemos es definir palabras nuevas . Dónde habremos leido eso antes !) Por lo tanto, lo que viene a continuación son un par de ejemplos mios que definen un par de palabras, que por una casualidad del destino sirven para hacer lo mismo. En realidad, son dos nuevas versiones de una palabra definida en el capitulo II (LIT#, ver ese capitulo para mas detalles!

Antes de continuar debe dejar claro un concepto muy importante, la pila es STACK D2.W siempre contiene el TOS, y A3.L siempre apunta al 2OS y se comporta como una pila Un ejemplo

```
move.w     d2, -(a3)
move.w     #3, d2        hemos introducido 3 en el stack , el anterior TOS es ahora el

move.w     (a3)+,d2      simplemente hemos hecho un DROP
```

A Me usado el Assembler de Metacomco por su facilidad de crear macros.

```
marker          equ       $4AFB              Ta mark the start of a code word
*
* macros
*
next            macro                        Fetches and executes the next
                move.w    (a1)+,d1           SUPERFORTH word, must be executed
                movea.w   0(a2,d1.w),a5      at the end of a code word
                jmp       0(a2,a5.w)
                endm
*
*
code            macro                        Creates a header
                dc.w      marker             Marks the start of a word
                dc.b      strg_end@-strg_start@     Gives the string length
```

```
prg_start#1     dc.b    -1                  The primitive's name
prg_end#1       equ     *
                cnop    0,2                 Pads to an even address
                dc.w    *2-*-2              Gives the code length
                endm
*
end_file        macro
                dc.w    0                   Must be placed at end of the code
                endm
*
*
*****************************************************
* Dos ejemplos de definiciones en Código Máquina *
*****************************************************
*
                code    'mmm ,mmm_end
                move.w  (a3)+,d0            20S en d0. 30S se convierte en 20S
                move.w  (a3)+,d1            20S en d1
                mulu    d2,d0               Multiplicamos TOS  por d0. Resultado en d0
                swap    d0                  cambio de palabras
                mulu    d2,d1               Multiplicamos TOS por d1  Resultado en d1
                add.l   d0,d1               Sumamos las palabras largas de d0 y d1.
                move.w  d1,-(a3)            ponemos palabra baja en 20S
                swap    d1
                move.w  d1,d2               ponemos palabra alta en TOS
                next


mmm_end


                code    'mm ,m_end
                move.l  (a3)+,d3            ponemos nuestro número de 32 b.t en d3
                tst.w   d2                  comprobar si TOS es cero
                beq     cero                si lo es salta a cero
                subq.w  #1,d2               restamos uno al TOS
                move.l  d3,d1               contenido de d1 igual que en d3
                bra.s   calcular
bucle
                add.l   d1,d3
calcular
                dbf     d2,bucle            si TOS es igual a cero no hay  bucle
                move.w  d3,-(a3)            ponemos palabra baja en 20S
                swap    d3
```

```
                move.w   d3,d2              ponemos palabra alta en f)S
                ora.b    #echo
cero
                move.w   #0, (a3)           Resultado es cero
                move.w   #0,d2

hecho
                rts1


m_end
                end file

                end
```

Junto a este articulo he enviado a la libreria los ficheros ejemplo0_asm, ejemplo1_asm ejemplo0_bin (UUUU) y ejemplo1_bin (UU).

Para cargar este solamente tenemos que teclear @ LOAD LOAD_BIN FLP1_EJEMPLO0_BIN

JUUU se ha definido igual que en el capitulo II, es otras palabras que se ha usado el metodo de multiplicar.

MU se ha definido por el sistema clásico, la cual

Como estoy enterado de que los capitulos I y II se perdieron misteriosamente en Olave es verdad, no lo comprendo, pues algunos de los articulos que envie se publicaron en el boletin de Octubre, y después dicen que hacen falta colaboraciones (ocho paginas de 45 lineas sin publicar)), voy a recordar que estas dos palabras sirven para multiplicar números de 32 bit por números de 6 bit, y el resultado es un número de 32 bit (naturalmente se trabaja sin signo y existen diferencias)

En el capitulo III escribi que la palabra MUUU no estaba definida en el manual pues esa comparado con la cantidad de palabras que se ha comido Digital Precision es una gota de agua en medio del oceano. A continuación un volcado de palabras que existen, pero por alguna razón extraña no han sido definidas en el manual

```
IPC          addr -- ) Llamada al IPC (intelligent peripheral controller  que lee el
     teclado  controla el sonido,etc ) addr apunta al comando del IPC  NO USAR A MENOS
     QUE ESTEMOS SEGURO DE LO QUE ESTAMOS HACIENDO.
TRAP1        , d1 n1 addr  --- d2 n2 )  Llamada ODOS TRAP 1,  los registros son cargados
     desde el stack:
          D1 L  es cargado con d1
          D0 W  es cargado con n1
          A3 W  y D2.W son cargados con addr (forth address ).
          d2 es el contenido de D1 L
```

```
          n2 es el contenido de D2.B

TRAP2       d1 a1 n2 --- d2 n3 ) LLamada ODOS TRAP 2, Los registros son cargados desde
     el stack:
          A0.L es cargado con d1 (dirección absoluta)
          D3.W es cargado con n1
          D0.W es cargado con n2
después del TRAP
          d2 es el contenido de A0.L
          n3 es el contenido de D0 (código error)

TRAP3       addr n1 d n2 --- n3 n4 n5 ) LLamada ODOS TRAP 3, los registros cargados son

          A1.W con addr ( FORTH address)
          D0.W con n1
          A0 L con 0
          D2.W con n2
          Después del TRAP
          n3 es el contenido de A1.W ( FORTH address )
          n4 es el contenido de D1.W
          n5 es el contenido de D0.W ( el código de error )

 TRAP3#       addr n1   d1 d2  n2 ---  n3 n4  n5 )  Como TRAP3,  pero con  el parametro
     adicional d2 , que es cargado dentro del registro D1.L

 VEC_UT       addr d1 d2 n1 --- d3 n2  ) LLamada una utilidad por vector, los registros
     cargados son:
          A1.W con addr   ( FORTH address )
          A0 L con d1
          D1.L con d2
          n1 es el vector número.
          Después la llamada el ODOS
          d3 es el contenido de A0 L
          n2 es el contenido de D0.W ( el código error )
```

Todas estas palabras han sido encontradas definidas en el FORTH 79 del mismo autor, que es una versión reducida del SUPERFORTH comercializado a bajo precio por mediación de QUANTA el D gita. C fue en un primer momento comercializado a finales de 1987 por Quanta con el nombre de SMALL-C) Pero si solamente fueran esas podríamos estar tranquilos, pero existen muchas más, que son: CALL_VU, CALL3RA, CALL3P, CALL_MT, GRAPHICS [_MT, LD_JOB FS_LOAD, *PSTR, >_S, V_ok, ZAP, ER, SL, PK, SH, BP,...

Por lo menos, yo no las tengo definidas en mi manual. No sé si, en un carta no creo que me contesten toda esa barrada) o coger un monitor para verlo todo y pasar el SUPERFORTH.

En el capítulo V, sobre matrices, se ha cometido un pequeño error. Pues creo que me he equivocado más propias matrices. Una matriz tiene n elementos, si con ambos ejemplos sería n+1, y ya contaba el cero como uno extra, mal (mis matrices tenían un elemento más porque ya contaba el cero). La solución es sencilla, y ahorramos memoria y es más rápido

En matrices de una dimensión en CREATE eliminamos 1+.

En matrices de dos dimensiones en CREATE eliminamos . 1+ . 1+ SWAP 1+ * .

El resultado es una matriz más clásica que la mía original. De todas formas envío junto este artículo un nuevo fichero de matrices.

En el último momento he pensado que sería mejor incluir una idea para solucionar el problema de los 55.374 bytes libres para el usuario en el diccionario. La idea es usar el mismo método de ahorrar espacio del autor del Superforth, almacenar rutinas en código máquina fuera del diccionario en forma de extensiones al Código a saltos a rutinas con retorno.

Naturalmente creo que esos bytes son suficientes para muchas aplicaciones y rara vez el usuario tendrá que hacer uso de ese truco, pues podrá usar la multitarea. Aunque lo lógico en una máquina de 32 bits es tener un FORTH STANDARD 32 Bits.

Este artículo ha sido terminado el día 1 de marzo de 1988. No creo que existan nuevos capítulos de este tamaño.

SALVADOR MERINO
FUENGIROLA (MALAGA-154

FORTH

Comandos gráficos en 3 dimensiones (I)

Tal y como es promet el mes pasado, voy a mostraros algunas palabras Forth que he escr to para un programa de dibujo en 3 dimensiones y que pueden ser ut l zadas como expresiones del lenguaje en cualquier otro programa. Me gustaría resaltar que es e es uno de los aspectos más bellos de este lenguaje su capacidad de amp l arse de un modo cas nf nito, adaptándose perfectamente a las necesidades de cada usuar o y de cada uso.

Pasando a describir el funcionamiento de estas nuevas palabras d re que para crear la mpresión de la tercera dimensión sobre la pantalla plana se basan en el uso de unos pocos princ p os básicos de perspectiva. Sustancialmente el mismo consiste en hacer que las rectas paralelas converjan en un punto de fuga; para ello se realizan unos cuentos cálculos bastante elementales, que permiten transformar cada grupo de 3 coordenadas X Y,Z en sólo 2 coordenadas (X,Y) representables en la pantalla.

Por otro lado se realiza también un cambio de coordenadas que s túa el or gen de los ejes X Y y Z en el centro de la pantalla. Con respecto al eje Z el origen se encuentr teóricamente 100 unidades detrás de la superficie de la pantal a; intentar s tuar un punt fuera de este por medio de una c nfirmac n inferior a 100 provocaría la aparic ón de un mensaje de error.

```
: COORD3D   ( x y z -- y x3D ) ( calcula una coordenada en 3D
            ROT 180 * SWAP 100 + / ;

: PUNTO3D   ( x y z -- x3D y3D )
            ( calcula las coordenadas de un punto en 3D)
            DUP   R  ( pasa a la pila de retornos una copia de z
            COORD3D 300 +  ( nueva eje de las x)
            R> ( recupera z)
            COORD3D   50 +  ( nueva eje de las y)  ;

: PUNTOF    ( x y -- Fx Fy )
            ( pasa a coma flotante las coordenadas de un punto)
            >R   FLOAT   R>   FLOAT   ;

: LINEA3D   ( x1 y1 z1 x2 y2 z2 -- ) ( dibuja una línea en 3D)
            >R  >R  >R ( pasa a la pila de retornos x2 y2 z2)
            PUNTO3D   PUNTOF
            R>  R>  R>  ( recupera x2 y2 z2)
            PUNTO3D   PUNTOF        LINE ;
```

Como se puede ver por los comentarios incluidos en el código fuente, la palabra COORD3D (léase coord-tres-de) realiza los cálculos de transformación de una coordenada a su equivalente en la representación en perspectiva.

PUNTO3D (léase punto-tres-de), basándose en COORD3D, calcula las dos coordenadas correspondientes a las tres originales de un punto en 3 dimensiones.

PUNTOF (léase punto-efe) se encarga de pasar a formato de coma flotante los valores de las dos coordenadas de un punto. Esta operación es necesaria por ser este el formato requerido por el GDOS para los comandos gráficos.

Finalmente LINEA3D (léase linea-tres-de) dibuja una linea en perspectiva sobre la pantalla plana a partir de las coordenadas de dos puntos situados en un espacio tridimensional.

PUNTO3D Y LINEA3D constituyen el auténtico corazón de un sistema de representación en perspectiva.

José Carlos de Prada
MADRID (CLave 216)

ORDENACION Y BUSQUEDA DE DATOS (III)

En el anterior episodio, vimos como la maligna presencia de un tiempo de ordenación función n cuadrado, hacía que todos nuestros esfuerzos en busca de un procedimiento rápido de ordenación fueran bastante valdíos, ya que la presencia de este maligno factor no se apartaba de los procedimientos de ordenación por intercambio, inserción o selección.

En esta segunda parte vamos a describir brevemente dos métodos de ordenación sustancialmente rápidos se trata de los métodos Shell y Quicksort

De estos el que puede llegar a ser más rápido es el Quicksort, pero el que en general en cualquier caso es más rápido es el Shell. En ambos métodos se precisa un poco de intuición por parte de las musas del programador, pues de fallar este el resultado volverá por los derroteros de n al cuadrado.

Ordenación por el método Shell

Se llama así en honor a su inventor D L Shell, aunque el nombre parece tener que ver con las conchas marinas (Shell) en Inglés que era decir concha. Dar a de la ordenación por inserción, en función de incrementos decrecientes. Lo que se hace es ordenar primero los elementos separados una distancia entre sí luego en sucesivas pasadas vamos disminuyendo esta separación hasta en una última pasada ordenar los elementos adyacentes. El aumento de eficiencia estriba en que en una pasada salvo en la última se intercambian pocos elementos en la ordenación y en todas se va incrementando el orden de los datos, normalmente el número de intercambios en la última pasada serán mínimos. Pudiera parecer que esto no fuera cierto pero de hecho es que lo hace

```
DEFine PROCedure Shell
largo=DIM matriz(1)
salto=largo DIV 2
REPeat ciclo1
        IF salto<=0 EXIT ciclo1
        FOR i=salto TO largo
                j=i-salto
                REPeat ciclo2
                        IF j<0 OR matriz(j)<=matriz(j+salto) EXIT ciclo2
                        t=matriz(j)
                        matriz(j)=matriz(j+salto)
                        matriz(j+salto)=t
                        j=j-salto
                END REPeat ciclo2
        END FOR i
        salto=salto DIV 2
        END REPeat ciclo1
END DEFine Shell
```

Esta rutina es exactamente lo que hace es en un bucle ex erno ir generando se n s an as de comparac ón en función de la dimensión de la matriz tomando cada vez la mi ad de la d stanc a anterior Pero existe un problema con la ordenacion shell Cuando la secuencia de ordenación toma valores múltiplos de 2 tiende , por oscuros mecan smos matemát cos a ser menos ef ciente Así que nada nos impide generar a gusto de usuario la secuenc a de sa to en las comparac ones, sabiendo que el último ha de ser 1 que cuantos más sa tos demos menos trabajo tendrá que hacer el último bucle

Siguiendo esta última propuesta y basándonos en la flexibi dad de los bucles FOR de SuperBASIC tenemos un Shell de aquesta guisa:

```
DEF PROCedure Shell_bis
largo=0 MEN matriz,1
FOR salto=9 5,3,2,1
        FOR pasalto TO largo
                pivote=salto
                REPeat ciclo
                        IF j<0 OR matriz(j)<=matriz(j+salto) EXIT ciclo2
                        temporal=
                        matriz(j)=matriz(j+salto)
                        matriz(j+salto)=temporal
                        j=j-salto
                END REPeat ciclo
        END FOR p
END FOR salto
END DEFine Shell_bis
```

Aquí definimos una secuencia propia de saltos, 9,5,3,2,1 aunque dicha secuencia habrá que hacerla en función de los datos a ordenar Como se ve no hay ningún número potencia de 2 Otra solución sería hacer el bucle de las mitades como en el caso anterior pero siempre tomando el número impar adyacente a la cifra correspondiente y así evitar las potencias de 2

En general y por estudios matemáticos fuera de éste lugar La ordenación shell tiene un tiempo de ejecución factor de n a la 1,2 .

Este factor ya es bastante más razonable pero se debe de tener en cuenta que con un apropiado análisis la ordenación Quicksort puede ser incluso más rápida

Ordenación por el método QUICKSORT:

Inventado por C A R Hoare, se le considera el mejor método de ordenación de todos los días Se basa en la ordenación por el método de intercambio, pero las burbujas se desplazan de nuevo La diferencia es que divide la matriz en dos trozos en función de un elemento más o menos central que denomina "comparando".

[illegible] empiezo tenemos la matriz [illegible] de acuerdo al elemento 1 [illegible] comparando en [illegible] en un principio paso lo dejaré propia el array en [illegible] la primera mitad con los menores del comparando y la segunda con los mayores o iguales

La selección del comparando se puede obtener aleatoriamente, y obtendrá la mediana de un grupo de elementos de la matriz. Para un ordenamiento ótimo el [illegible] elemento mediano aunque [illegible] en el caso peor. Quicksort según [illegible] funciones aunque [illegible] termina la burbuja [illegible]

La siguiente versión de Quicksort elige el elemento central de la matriz lo cual siempre puede resultar en una buena elección aunque es una técnica tan simple que funciona.

```
DEFine PROCedure Quicksort
largo=DIM(matriz())
Qs 1,largo
END DEFine Quicksort

DEFine PROCedure Qs (izq,der)
 x = matriz((izq+der)/2)
REPeat ciclo1
          REPeat ciclo2
          IF matriz(i)<x: DO EXIT ciclo2
          i=i+1
          END REPeat ciclo2
          REPeat ciclo3
          IF x<matriz(j): DO EXIT ciclo3
          j=j-1
          END REPeat ciclo3
IF i<=j
          SWAP matriz(i),
          matriz$ = matriz(i)
          matriz(i) = matriz(j)
          matriz(j) = matriz$
          i=i+1 : j=j-1
END IF
 EXIT izq>der OR i>der
END REPeat ciclo1
 IF izq<j: Qs izq, j
IF i<der: Qs i, der
END DEFine Qs
```

La desventaja de las comparaciones e intercambios es compleja pero se puede [illegible] número de comparaciones función de n longitud de la matriz es de [illegible] número de intercambios de n/6 log n la cual es sensiblemente menor que [illegible] antes considerado

Así para ordenar 100 elementos se utilizan 100*100=10000 comparaciones contra las 930 del método de la burbuja. Hay que tener en cuenta que todo esto depende de la elección del comparando y que en el peor de los casos será tan burbuja. Detalle que no ocurre nunca con la ordenación shell que es en sí misma un método de dos. Pero normalmente la elección del comparando se puede obtener fácil y efectivamente en función de la naturaleza intrínseca de los datos con los que se va a operar por el programador y en un estudio de edades del censo electoral no se tomará como comparando para su ordenación el valor 1 y el valor 100 lo lógico sería tomar un valor entre 45 y 55 y según el país de que se trate.

Bien ya hemos cogido parte de los objetivos de esta serie de artículos. En sus próximos estudiaremos la ordenar ón de otros tipos de datos y de datos en fichero así como la búsqueda de datos ordenados

Hasta Entonces
QUASI DIOS
Valladolid

P.D: Decirte que los procedimientos de esta serie así como su esquema general van siguiendo el que aparece en el libro de Herbert Schildt "Lenguaje C programación avanzada" editado por Osborne McGraw-Hill en español por si alguna vez quieres seguir con algo más de profundidad

PROCEDIMIENTOS DE ORDENACIÓN

El proceso de datos es desde siempre una de las utilizaciones más comunes de l informática. El proceso de datos podemos definirlo como el almacenamiento manipulación y recuperación de información. Cuando hablamos de manipulación pensamos en ordenaciones búsquedas y refundiciones de archivos de datos. Por esta razón se puede decir que casi no queda nada por descubrir en estos campos. Estas páginas son una recopilación de los procedimientos más comunes de ordenación de archivos soportados en una estructura llamada de tabla.

Estructuras de datos más comunes.

Normalmente la información de los archivos se divide por registros. Cada registro trata de un elemento diferencial del archivo y se divide en campos. El campo que origina la ordenación es la clave de la información. Por ejemplo podemos definir un archivo de socios de Olave con tres campos, número de socio, nombre y dirección. Podemos ordenar el archivo por orden creciente de número de socio y en ese caso el campo del número de socio será la clave de ordenación y los demás campos la información asociada.

Una estructura de tabla es aquella en que los elementos del archivo son todos accesibles directamente. Una matriz de variables tiene la estructura de tabla. Para ordenar una tabla hay que situar de forma creciente y decreciente sus elementos de forma que sus claves queden en orden creciente.

Otra estructura muy usada es la de lista encadenada. Se caracteriza porque se puede ordenar sin desplazar sus elementos. Esto es debido que utiliza algun campo de información para definir la situación de cada elemento dentro de la lista, es decir, cual es el anterior y cual el siguiente. Una lista, por definición, se crea ordenada. Si se comienza de cero se crea una lista vacia y se van añadiendo los elementos. Los elementos se insertan en la lista modificando la información de la situación del anterior y del posterior. Una lista ocupa más memoria que una tabla pero es más facil de actualizar. Archivo utiliza una estructura de lista para sus archivos.

Como funcionan los procedimientos.

Los procedimientos que veremos están pensados para que sean capaces de ordenar tablas numéricas de hasta dos dimensiones y alfanuméricas hasta de tres. La primera dimensión nos indica el número de elementos, la segunda el número de campos y la tercera en las alfanuméricas, la longitud máxima de las cadenas. Se pueden ordenar según diferentes claves y elementos concretos de las matrices dentro de dos elementos dados. Para ello todos los procedimientos necesitan cierto parametros que son por este orden: Nombre de la matriz a ordenar, nombre de la matriz que se usará como auxiliar, primer elemento a ordenar, último elemento a ordenar, primer campo utilizable de la matriz, último campo utilizable de la matriz y número del campo que es

La matriz auxiliar es una de características idénticas a la que se va a ordenar excepto cuando que tiene una dimensión menos, la primera. Por ejemplo si queremos ordenar la matriz numeros$ (10,20), deberemos dimensionar, antes de llamar al procedimiento de ordenación que queramos, otra auxiliar de nombre, por ejemplo, nombre_bis$ y dimensiones 10,20 Es decir nombre bis$(10,20)

El primer y último elementos a ordenar son por si queremos ordenar únicamente una parte de una matriz que será la comprendida entre estos dos números, incluyendolos

Se necesita el número del primer y último campo utilizable porque las matrices en SUPERBASIC tienen campo cero, que a veces no se utiliza, y para saber el valor de la segunda dimensión de la matriz a ordenar

Los procedimientos utilizan otros dos auxiliares que sirven para contrarrestar la falta en SUPERBASIC de la instrucción SWAP que intercambia los valores de dos variables, y la imposibilidad de pasar la información de una matriz a otra con una sola sentencia de asignación como matriz_a=matriz_b

```
100 DEFine PROCedure cambiar(primero,segundo,salvaguarda,dim1,dim2
110   iguala primero,salvaguarda,dim1,dim2
120   iguala segundo,primero,dim1,dim2
130   iguala salvaguarda,segundo,dim1,dim2
140 END DEFine cambiar
150 DEFine PROCedure iguala(primero,segundo,dim1,dim2)
160   LOCAL a
170   IF dim2=0 THEN
180     segundo=primero
190   ELSE
200     FOR a=dim1 TO dim2:segundo(a)=primero(a)
210   END IF
220 END DEFine iguala
```

Por ejemplo, imaginemos que tenemos los datos de todos los integrantes de CLAVE en un archivo de tres campos, como hemos explicado anteriormente. La matriz que sostiene el archivo, imaginemos que sea socios$(1000,3,20). Recordábamos que tenía tres campos, entonces como dimensionamos tres no usamos el primero. Si queremos ordenar este archivo entero por número de socio por el método de burbuja deberemos hacer:

```
DIM socios_bis$(3,20):burbuja socios$,socios_bis$,1 1000,1,3,1
```

Si utilizamos el campo cero la matriz socios$ sería socios$(1000,2,20). Si queremos ordenar los últimos 200 elementos de este archivo por el método Shell deberemos hacer

```
DIM socios_bis$(2,20):shell socios$,socios_bis$,800,1000,0,2,0
```

Caracteristicas de los procedimientos de ordenación.

Primeramente todos los procedimientos de ordenación se pueden escribir de dos formas diferentes iterativa y recursiva. La manera iterativa consiste basicamente en la repetición de un número determinado de instrucciones hasta que se consigue un fin. Los procedimientos recursivos se llaman a si mismos empleando un proceso cíclico. La recursividad solo estará presente en lenguajes estructurados avanzados (en el SUPERBASIC la tenemos).

De los ocho procedimientos representativos recorridos en estas páginas el Rápido y Extracción están escritos en su versión recursiva. Normalmente el Rápido es así como se suele escribir.

Para ser preciso al hablar de los procedimientos de ordenación y su mayor o menor valor hay que explicar algunos factores de las ordenaciones que influyen de manera importante.

En primer lugar el número de elementos a ordenar por varias razones. Porque existen procedimientos excelentes para pocos elementos y malos para muchos. Y porque normalmente al número de elementos a ordenar multiplica el tiempo de ejecución de forma muy superior a su crecimiento.

En segundo lugar determinar si la tabla está ya parcialmente ordenada o no. Algunos procedimientos reducen mucho sus tiempos cuando se dá este caso.

La regularidad no es la misma en todos los procedimientos. Regularidad es la capacidad de un procedimiento de tardar el mismo tiempo de ejecución para clasificar el mismo número de elementos.

Por último la estabilidad de procedimiento puede interesarnos. Se dice que un procedimiento es estable cuando dos elementos con una clave igual no trocan sus posiciones iniciales al final de la clasificación.

Pruebas realizadas.

Para tener una idea precisa de sus características he pasado todos estos procedimientos por un banco de pruebas consistente en cronometrar sus tiempos de ejecución variando tres factores, el número de elementos, el nivel de ordenación existente en la tabla a ordenar y la tabla misma. Para ello he hecho las pruebas con tablas de números enteros de entre cinco y mil elementos. Son registros de un solo campo que es a la vez clave e información. No es ni más ni menos que ordenar una lista de números.

Primero: ONDULANTE

El nombre 'Ondulante' es una castellanización del nombre inglés Rapid que significa rápido, ondulación y que como veremos es el más apropiado a la vista de su funcionamiento. Es un procedimiento ya viejo y consiste basicamente en ir comparando sucesivamente los

elemen ni ques de la tabla, comenzando desde el primero, e no rt endo os es ecesar a 1 nal de la primera pasada (como una ola y onda/ el mas grande es el a 1 nal de a tabla. En ondas sucesivas se iran colocando progresivamente los restan es a ordenación termina cuando un testigo nos delata, despues de una ola" que no ha hab do nversiones o cuando se haya dado un numero de pasadas igual al número de elemen os de a tabla s en cada pasada colocamos uno, en las N pasadas colocaremos los N elementos

Este proced m ento es muy corto de escribir pero es el más lento de los ocho El t empo de e ecuc ón se multiplica exageradamente al aumentar el número de e mentos de la tabla. No se debe usar cuando hay más de treinta elementos pues se hace lent s mo

El proced miento es estable y bastante regular. Cuando la tabla esta semi-o denada se reduce mucho el tiempo de ejecución, no tanto como otros proced m entos como Insercion D Shell,-Metsner, pero si más que los demas.

Resumiendo

Sencillo
Estable
Bastante regularidad.
Se acerca mucho con tablas semi-ordenadas.

N elementos	5	10	20	30	50	75	100	200	300
tiempos en sg	56	2 33	9 26	21	61	134	249	981	2161

El lgoritmo del procedimiento es.

```
230 DEFine PROCedure ondulante(t(),t_dis,comienzo,final,dim1,dim2 clave
240   LOCal onda,origen,testigo
250   FOR or genecomienzo TO final
      test.go=0
270     FOR onda=final-1 TO origen STEP -1
280       IF t(onda+1,clave)<t(onda,clave) THEN
290         cambiar t(onda),t(onda+1),t_dis(dim2),dim1,dim2,testigo=1
300       END IF
310     END FOR onda
320     IF testigo=0 THEN EXIT origen
330   END FOR origen
340 END DEFine ondulante
```

Segundo BURBUJA

Este proced m ento es de los mas conocidos. Su nombre ingles es Bubble En algún libro consultado se llamaban ordenación por Intercambio. Es muy similar al Ondulante y sus prestaciones son parecidas

Consiste en ir comparando cada elemento, comenzando por el primero con los restantes e invirtiendolos si hace falta.

Despues de la primera pasada el elemento mas pequeño estará al comienzo de la tabla después de ascender o flotar, de ahí el nombre de la variable como una burbuja en una botella de agua hasta su posición. Despues de N-1 pasadas estarán situados correctamente los N elementos de la tabla. Este procedimiento no puede ser detenido antes de las N-1 pasadas pues no tiene testigo de inversión. Otra diferencia con el método anterior es que aquel comparaba elementos de posiciones consecutivas que iban descendiendo como una a una en este el elemento que compara está fijo.

Al igual que el procedimiento anterior es estable y muy regular con tablas ya ordenadas pese a que disminuye sensiblemente su duración con tablas semiordenadas.

Resumiendo:

Muy sencillo.
Muy regular
Estable.
Se acerca bastante con tablas semi-ordenadas.

N elementos	5	10	20	30	50	75	100	200	300
tiempo en sg	.55	2.26	9 13	21 1	59 2	130	241	948	2111

El listado es:

```
850 DEFine PROCedure burbuja(t,t_bis,comienzo,final,dim1,dim2,clave
860   LOCAL origen,flota
870   FOR origen=comienzo TO final-1
880     FOR flota=origen+1 TO final
890       IF t(flota,clave)<t(origen,clave)THEN
900         cambiar t(origen),t(flota),t_bis(dim2),dim1,dim2
910       END IF
920     END FOR flota
930   END FOR origen
940 END DEFine burbuja
```

Tercera: EXTRACCION

Este metodo tambien se llama "Permuta de indices" y "Por minimos" segun los autores que lo tratan. Como ya he dicho antes no escojido la version recursiva pues me parece más elegante. Se trata de encontrar el mínimo de la tabla y situarlo en el comienzo de la misma. Esto lo hace el procedimiento BUSCAR MINIMO. Esta operacion hay que ir haciendola repetidamente con el resto de la tabla hasta completarla.

Aunque parece el mismo, existen cuatro notorias diferencias con el método de Burbuja que hacen que sus prestaciones sean muy diferentes. Este es recursivo. Debido a la existencia de la variable mínimo no se intercambian los elementos hasta encontrar el que va

a su lugar definitivamente en la posición temporal por lo que se ahorran muchos intercambios. Por esta razón el número de comparaciones e intercambios es constante por que el procedimiento es muy regular y no esta influido por la posible ordenación de la tabla. El procedimiento es inestable ya que no se respeta el orden inicial de dos elementos con la misma clave. Hasta los cincuenta elementos es uno de los métodos más rápidos.

Resumiendo:
Sencillo
Muy regular
Inestable
No se acorta con tablas semi-ordenadas.

N elementos	5	10	20	30	50	75	100	200	300
tiempo	69 63	1 56	4 13	7 55	17 1	34 1	56 3	202	435

El listado de los dos procedimientos es

```
450 DEFine PROCedure buscar_minimo(t,t_bis,prin,fin,dim1,dim2,clave
460   LOCal origen,origen_bis,minimo
470   minimo=t(prin,clave):origen_bis=prin
480   FOR origen=prin+1 TO fin
490     IF t(origen,clave)<minimo THEN
492       minimo=t(origen,clave):origen_bis=origen
494     END IF
500   END FOR origen
510   cambiar t(prin),t(origen_bis),t_bis(dim2),dim1,dim2
520 END DEFine buscar_minimo
530 DEFine PROCedure extraccion(t,t_bis,comienzo,final,dim1,dim2,clave
540   IF final-comienzo>1 THEN
550     buscar_minimo t,t_bis,comienzo,final,dim1,dim2,clave
560     extraccion t,t_bis,comienzo+1,final,dim1,dim2,clave
570   END IF
580 END DEFine extraccion
```

Cuarto: INSERCION

Este método es uno de los más antiguos. Consiste en coger uno a uno los elementos de matriz desde el segundo a ordenar hasta el último e insertarlos en la parte de la tabla ya ordenada, que es la que está entre el primero y el que se coge. Hasta tablas de veinte elementos es un método de ordenación bastante rápido, además de uno de los más sencillos de escribir. En archivos grandes los tiempos se disparan pero menos que en los métodos Ondulante y Burbuja. En tablas aleatorias tiene una regularidad tirando a baja los tiempos de ordenación en estos casos varían hasta un treinta por ciento. Es mejor es estable.

Pero quizá lo más importante de este método de es la gran reducción de tiempo que experimentan las ordenaciones de tablas semi ordenadas, más que ningun otro por lo que

es que se utiliza universalmente para actualizar archivos. Resumiendo

Muy sencilla
Estable
Poca regular ded.
Enorme reducción de tiempos con archivos semi-ordenados.

N.e mentos	5	10	20	30	50	75	100	200	300
t mpo en sg	,61	1.76	5.56	11 6	30 1	63 2	119	434	952

El l strado es

```
590 DEF.ne PROCedure insercion(t,t_bis,comienzo,final,dim1,dim2,clave
600   LOCal inserto,desplazo,e
610   FOR insertо=comienzo+1 TO final
620      equ.ar t(insertо),t_bis(dim2),dim1,dim2
630     FOR desplazo=origen TO comienzo STEP -1
640       IF t_bis(clave)<t(desplazo-1,clave)THEN EXIT desplazo
650       .gualar t(desplazo-1),t(desplazo),dim1,dim2
660     END FOR desplazo
670     igualar t_bis(dim2),t(desplazo),dim1,dim2
680   END FOR insertо
690 END DEFine insercion
```

Quicksort SHELL

Este método lleva el nombre de su inventor, D. L. Shell. Se trata de una optimización de los métodos clásicos Ordenante, Burbuja e Inserción. Los métodos clás,cos tienen el defecto de que cada elemento a insertar debe ser comparado con cada uno de la parte de la matr.z ya ordenada para encontrar su lugar, y esta comparación avanza de uno en uno. Shell pensó, con acierto, que si en vez de uno en uno los saltos fueran mas largos se encontraría antes la posición definitiva. Es la misma idea desarrollada en las búsquedas binarias. Se trata de dividir la tabla o matriz inicial en subtablas de dos elementos separados por un intervalo dado. Primero se define un intervalo inicial que será igual a la mitad de la tabla. Este intervalo será reducido a la mitad en cada pasada hasta que sea menor que uno, momento en que terminará la ordenación. En cada pasada hacemos una clasificación de las subtablas existentes invirtiendo los elementos si es necesario. En la última pasada el intervalo será uno, es decir se comparará cada elemento con el siguiente, la tabla estará ya ordenada en lo fundamental, es decir, estarán dados los saltos más largos por lo que la ordenación será muy rápido.

Es un método medianamente rápido hasta que se superan los cien elementos, entonces se nota con claridad su superioridad con los anteriores. Quizás no sea muy fácil de entender pero el procedimiento no es largo de escribir.

Tiene una regularidad mediana, con variaciones de tiempos para archivos aleatorios de un veinte por ciento. Al igual que el de inserción reduce apreciablemente los tiempos con

```
arch dos semi ordenados, aunque no tanto  No es estable  Resumiendo

     Senc illo
     Medianamente regular.
      nestable
     Se reducen mucho los tiempos  en tablas semi ordenadas.

     N elementos     5     10    20    30    50    75    100   200   300
     tiempo en seg   .57   1 0   5 53  9 65  25    37.8  66 1  185   253

     El listado es:

     700 DEFine PROCedure shell(i,i_bis,comienzo,final,dim1,dim2,clave
     710   LOCal origen,intervalo,testigo,incremento,desviacion,uno dos
     720   intervalo=final-comienzo+1
     730   REPeat uno
     740     intervalo=INT(intervalo/2):IF intervalo<1 THEN EXIT uno
     750     REPeat dos
     760       testigo=0 incremento=final-intervalo
     770       FOR origen=comienzo TO incremento
     780         desviacion=origen+intervalo
     790         IF i(origen,clave)>i(desviacion clave) THEN
     800           cambiar i(origen),i(desviacion),i_bis(dim2),dim1,dim2
     805                 testigo=1
     810         END IF
     820       END FOR origen
     830       IF testigo=0 THEN EXIT dos
     840     END REPeat dos
     850   END REPeat uno
     860 END DEFine shell
```

Sentencia RAPIDA

La ordenación Rapida, llamada Quicksort en inglés, fue inventado por C. Hoare a principios de los años sesenta y mejorado mas tarde por R. Sedgewick. En principio resulta un poco chocante la falta de modestia de su inventor a la hora de escoger un nombre sobre todo teniendo en cuenta que, como veremos, hay metodos más rápidos Esta ordenación es verdad que es rápida para tablas grandes, sin embargo es lenta para tablas pequeñas Por este motivo el procedimiento que veremos no es "puro", tiene que es un híbrido que utiliza el procedimiento de insercion al final de la clasificación Este procedimiento para funcionar debe cargarse, además de con los procedimientos CAMBIAR y GUARDAR como los demás con el procedimiento INSERCION

El método consiste en elegir un elemento de la tabla cuya clave sea de un valor aproximado a la media y despues dividir la tabla en tres subconjuntos, el elemento

menor que llamaremos Pivot, los que tienen la clave inferior a Pivot a la que tienen superior. A continuación se explora la tabla del primero al último para ver si encontramos algún elemento de clave superior al Pivot y del último al primero para ver si encontramos otro de clave inferior. Si se da ese caso se intercambian los elementos en caso contrario dividiremos los subconjuntos de nuevo y haremos con los nuevos lo mismo cuando los subconjuntos se hagan tan pequeños que no alcancen a tener diez elementos entonces se deja de dividir y se efectua una ordenación por inserción de estos por la razón que había apuntada más arriba.

Este procedimiento consta de tres subprocedimientos propios que son EXPLORACION, ELEGIR_PIVOT y DIVISION que sirven para lo siguiente:

EXPLORACION - explora los elementos de los subtablas hasta encontrar dos que superan ese carácter si es necesarios para intercambiarse.

ELEGIR PIVOT - elige el pivot mas adecuado para el desarrollo de la ordenación Este debe ser aproximado a la media de las claves.

DIVISION - divide las subtablas cuando tienen mas de nueve elementos y las exploraciones se han encontrado sin hallar elementos susceptibles de ser intercambiados Este procedimiento se llama a si mismo, recursividad, hasta que hayan menos de nueve elementos y en ella está la razón de que sea recursivo este ejemplo de ordenación Rápida.

Esta ordenación consigue ser eficaz a partir de los treinta elementos A partir de los doscientos se destaca claramente con Shell-Metzner y Arbol por esta razón son los únicos procedimientos que he probado con tablas de hasta mil elementos

El método rapido es muy irregular y los tiempos llegan a variar en tablas aleatorias hasta un cuarenta por ciento Como se puede observar utiliza siete procedimientos que hacen un total de setenta líneas de programa, por lo que diremos que no es nada sencillo de escribir, además no es estable Curiosamente este procedimiento tiende a aumentar sus tiempos cuando trabaja con tablas semi-ordenadas

Resumiendo:

Muy aparatoso de escribir
Muy irregular
No es estable
No se reducen los tiempos con tablas semi-ordenadas sino que aumentan

N elementos	5	10	20	30	50	75	100	200	300	500	1000
tiempo en sg	66	2 13	5.9	9 5	19	31.6	49 5	111	206	431	1121

```
870 DEFine PROCedure exploracion(t,t_bis,prin,fin,dim1,dim2,clave
880   LOCal izq,dcha,uno,dos
890   izq=prin:dcha=fin+1
900   pivot=t(prin,clave)
910   REPeat uno
920     IF izq>=dcha THEN EXIT uno
930     REPeat dos:izq=izq+1:IF t(izq,clave)>=pivot THEN EXIT dos
```

```
940      REPeat dos dcha=dcha-1 UNTIL t(dcha,clave)<=pivot THEN EXIT dos
950        IF  izq<dcha THEN  cambiar t(izq) t(dcha),t_bis dim2 ,dim1 d m2
960    END REPeat uno
970    IF izq<dcha THEN cambiar t(izq),t(dcha),t_bis(dim2),dim1,dim2  izq= izq 1
980    lugar= izq.cambiar t(lugar),t(pria),t_bis(dim2),dim1,dim2
990 END DEFine exploracion
1000 DEFine PROCedure elegir_pivot(t,t_bis,izq,dcha,dim1,dim2 clave
1010   LOCal medio
1020   medio=INT((izq+dcha)/2)
1030   IF t(izq,clave)<=t(medio,clave  THEN
1040     IF t(medio,clave)<=t(dcha,clave) THEN
1050       pivot=medio
1060     ELSE
1070       IF t(dcha,clave)<=t(izq,clave) THEN pivot=izq ELSE pivot=dcha
1080     END IF
1090   ELSE
1100     IF t(dcha,clave)<=t(medio,clave) THEN
1110       pivot=medio
1120     ELSE
1130       IF t(izq,clave)<=t(dcha,clave) THEN pivot=izqELSE pivot=dcha
1140     END IF
1150   END IF
1160   cambiar t(izq),t(pivot),t_bis(dim2),dim1,dim2
1170 END DEFine elegir_pivot
1180 DEFine PROCedure division(t,t_bis,prin,fin,dim1,dim2,clave,
1190   IF fin-prin)=9 THEN
1200     elegir_pivot t,t_bis,prin,fin,dim1,dim2,clave
1210     exploracion t,t_bis,prin,fin,dim1,dim2,clave
1220     IF lugar-prin>fin-lugar THEN
1230       division t,t_bis,lugar+1,fin,dim1 dim2,clave
1240       division t,t_bis,prin,lugar-1,dim1,dim2,clave
1250     ELSE
1260       division t,t_bis,prin,lugar-1 dim1,dim2,clave
1270       division t,t_bis,lugar+1,fin,dim1,dim2,clave
1280     END IF
1290   END IF
1300 END DEFine division
1310 DEFine PROCedure rapida(t,t_bis,comienzo,final,dim1,dim2,clave
1320   division t,t_bis,comienzo,final,dim1,dim2,clave
1330   insercion t,t_bis,comienzo,final,dim1,dim2,clave
1340 END DEFine rapida
```

5i .mo. SHELL METZNER

Como su nombre indica es una version mejorada del metodo Shell. Las diferencias con el método de Shell son las siguientes:

Si en el método de Shell el salto máximo en cada pasada era igual al intervalo en el Metzner si dos elementos son cambiados de posicion y la longitud de la tabla lo permite se compara de nuevo el que desciende para ver si puede seguir dando saltos del mismo valor hasta encontrar su lugar

No existe testigo de inversion y no hace falta dar una ultima pasada de comparaciones sin hacer cambios para pasar a un intervalo menor, con lo que se ahorran muchas operaciones inutiles

Por todas estas razones reduce a un tercio el número de comparaciones con respecto al método de Shell y a la mitad sus tiempos de ejecucion El metodo es practicamente igual de sencillo de escribir que el metodo de Shell. Es bastante regular y reduce apreciablemente los tiempos de ejecución con tablas semi-ordenadas. Como defecto es igual que Shell es inestable.

El método de Shell-Metzner es, segun mis pruebas, el que menos se ve afectado por el incremento del número de elementos de todos. Es el más rapido con tablas pequeñas y grandes.

Por todo esto es probablemente el más logrado de todos los metodos de ordenación que comento en estas páginas. Resumiendo:

Sencillez mediana.
Regular.
Inestable
Reduce bastante los tiempos con tablas semi-ordenadas

N elementos	5	10	20	30	50	75	100	200	300	500 1000
tiempo en sg.	.47	1.4	4.19	6.85	16.9	24.3	43.1	113	150	302 749

Su listado es el siguiente:

```
1350 DEF.no PROCedure shell_metzner(t t_bis,comienzo,final,dim1,dim2,c,aux,
1360   LOCal origen,origen_bis,intervalo,incremento,desviacion,uno,dos
1370   intervalo=final-comienzo+1
1380   REPeat uno
1390     intervalo=INT(intervalo/2):IF intervalo<1 THEN EXIT uno
1400     incremento=final-intervalo
1410     FOR origen=comienzo TO incremento
1420       origen_bis=origen
1430       REPeat dos
1440         desviacion=origen_bis+intervalo
1450         IF t(origen_bis,clave)<=t(desviacion,clave)THEN EXIT dos
```

```
1460          cambiar (variable b.s ,'desviacion',' bis(dim2',d m1 d m2
1470          origen_bis=origen_bis-intervalo
1480          IF origen_bis<comienzo THEN EXIT dos
1490        END REPeat dos
1500      END FOR origen
1510    END REPeat uno
1520 END DEFine shell_metzner
```

Ordenar ARBOL

Esta clasificación es una de las más originales. Se basa al igual que el método de shell en el acortamiento de los pasos para trasladar un elemento de un lugar a otro de la tabla, al hacerse este traslado con saltos binarios. Para hacer esto se apoya en una estructura de ARBOL que da nombre al método. Una estructura de árbol es aquella en que los elementos de una tabla se organizan de una manera 'arborescente', es decir el elemento primero es la raiz y de ella parten dos ramas que son los dos elementos siguientes. De cada rama parten otras dos hasta completar todos los elementos de la matriz. Para verlo de una manera más clara una tabla de doce elementos se representaría en forma de árbol de la siguiente manera

```
                    1
                2           3
            4       5       6       7
          8   9   10  11  12
```

Esta estructura tiene la particularidad, por este motivo de ser q.r que el primer elemento a ordenar debe ser siempre el número 1. Se puede observar que si el nudo o raiz de una rama es el elemento x sus hojas izquierda y derecha serán los elementos 2x y 2x+1. Básicamente la ordenación Arbol es buscar el elemento mas grande del arbol y hacerlo subir desde su rama hasta la raiz del arbol (elemento número 1) para luego enterrarlo intercambiandolo con el último elemento. Después se repetirá esta operación sucesivamente con las cuartetas (1,n-1), (1,n-2), etc hasta llegar a la tabla 1 1. El procedimiento principal usa otros tres secundarios que son:

MAXIMAL - sirve para ordenar un subarbol de manera que la raiz tenga la clave de ordenación superior a la de cualquiera de las ramas y esto se llama hacerlo maximal. Para ello primero busca cual es la mayor de las ramas y después, si hace falta lo intercambia con la raiz.

INICIALIZACION.- sirve para recorrer cada uno de los subarboles haciendolos maximales.

ORDEN_MAX.- sirve para extraer el elemento que ha subido a la raiz del arbol, intercambiarlo con el último y recomenzar de nuevo con la tabla menos este elemento.

ARBOL.- es el procedimiento que apoyandose en los anteriores efectua la ordenación. Como se vé antes, para no pararse enviando un mensaje de error, que el primer elemento a ordenar tenga subindice uno. Este procedimiento empieza a dar buenos resultados a partir de

los dos ordenados elementos y se observa que va en progresión. No lo he probado con más de mil elementos pero es posible que sea el más rápido.

No es sencillo de escribir y menos de entender. Su regularidad es total, siempre ayuda el mismo tiempo con el mismo número de elementos, y no le afecta si la tabla está semi ordenada. Es estable. Resumiendo:

Complicado de escribir.
Extremadamente regular.
Estable.
No le afecta que la tabla esté semi ordenada.

N elementos	5	10	20	30	50	75	100	200	300	500	1000
tiempo en sg.	2	4.5	9.5	15.5	27.5	43.5	61.5	137	217	987	854

Su código es:

```
1530 DEF np PROCedure maximal(t,t_bis,prin,fin,dim1,dim2,clave)
1540   LOCal arbol_max,raiz,indice,ene,e
1550   raiz=prin:arbol_max=0
1560   REMark uno
1570   REPeat uno
1580     IF (2*raiz)>fin OR (arbol_max) THEN EXIT uno
1590     ene=2*raiz
1600     IF 2*raiz+1<=fin THEN
1610       IF t(2*raiz+1,clave)>t(2*raiz,clave) THEN indice=ene+1
1620     END IF
1630     IF t(indice,clave)>t_bis(clave) THEN
1640       POKE@ t(indice),t(raiz),dim1,dim2
1650       raiz=indice
1660     ELSE
1670       arbol_max=1
1680     END IF
1690   END REPeat uno
1700   POKE@ t_bis(dim2),t(raiz),dim1,dim2
1710 END DEFine maximal
1720 DEFine PROCedure montonizacion(t,t_bis,fin,dim1,dim2,clave)
1730   LOCal origen
1740   FOR origen=INT(fin/2) TO 1 STEP -1
1750     maximal t,t_bis,origen,fin,dim1,dim2,clave
1760   END FOR origen
1770 END DEFine montonizacion
1780 DEFine PROCedure orden_max(t,t_bis,dim1,dim2,clave)
```

```
1 ?0    LOCAL origen,una
1000    origen=final_bis
1010    REPeat una
1020      IF origen<=comienzo bis THEN EXIT una
1030      camb mt t(comienzo bis) t(origen),t bis(dim2),dim1,dim2
1040      origen=origen-1
1050      maximal t,t bis,comienzo bis,origen,dim1,dim2,clave
1060    END REPeat una
1070  END DEF no orden_max
1080  DEF no PROCedure arbol(t,t bis,comienzo,final,dim1,dim2,clave
1090    comienzo_bis=comienzo:final_bis=final
1900    IF comienzo_bis<1 THEN
1910       no_recursion t,t_bis,final_bis,dim1,dim2,clave
1920      orden_max t,t_bis,dim1,dim2,clave
1930    ELSE PRINT"ERROR EN LOS INDICES DE LA TABLA"
1940    END IF
1950  END DEFine
```

Resumen final.

Finalmente y segun hemos visto se pueden sacar una serie de conclusiones.

Si queremos ordenar una tabla completamente desordenada de menos de mil elementos deberemos usar el método de Shell-metzner.

Si la tabla desordenada tiene mas de mil elementos el método de Arbol tambien es bueno

Si queremos ordenar una tabla que ya está ordenada en su mayoria el método de inserción es el más rápido

Resumen de tiempos para tablas desordenadas de un solo campo.

N elementos		5	10	20	30	50	75	100	200	300	500	1000
ONDULANTE	56	2 39	9.36	21.7	61	134	249	981	2181			
BURBUJA	55	2 26	9.13	21 1	59.2	130	241	948	2111			
EXTRACCION		.63	1.56	4.13	7.55	17.1	34.1	96 5	202	438		
INSERCION	.61	1 26	5.56	11.6	30.1	63.2	115	424	952			
SHELL	57	1 8	5.59	9 65	25	37.9	66.1	189	259			
RAPIDA	.66	2 13	5.3	9 5	19	31.6	49.5	111	208	433	1121	
SHELL-METZNER		.47	1.4	4.13	6.85	16.9	24 2	43 1	113	150	302	745
ARBOL	2	4.5	9.5	14.5	27.5	42.5	61 5	137	217	387	854	

En los gráficos adjuntos los números verticales de la izquierda es el tiempo en segundos, los horizontales de la parte superior son el número de elementos a ordenar las letras de la parte superior y derecha las iniciales de los métodos de ordenación

O Ondulante
B Burbuja
E Extracción
I Inserción
S Shell
R rápida
M Shell Meizner
A Arbol

Bibliografía.

Debido a razones de espacio no se puede estudiar completamente los procedimientos tratados. Ou rás existia alguno que no haya quedado muy claro Si soc o de CLAVE que desee conocer más profundamente todo lo relacionado con el tema puede consultar, entre otros los siguientes libros:

Procedimientos de clasificación R Gurher MASSON
El basic y sus ficheros Jacques Boisgontier ELISA (2 libros
Programación avanzada en basic Peter Bishop ANAYA MULTIMEDIA

Existen otros muchos libros que explican o contienen listados de procedimientos de clasificación, pero la mayoria, lamentablemente, habla de los métodos de Burbuja e Inserción que son de los peores como hemos visto.

Jorge Diaz
Grenta Clave 139

El programa VENTANAS, contiene en fichero (VENTANAS EXT) que incorpora una nueva función al BASIC (WINDOW VARS, cuando es cargado en memoria de la forma habitual

```
10 a=RESPA(150)
20 LBYTES modul_ventanas_ext,a
30 CALL a
```

Una vez cargado, se puede la función puede ser llamada mediante

```
base=WINDOW_VARS(canal)
```

donde canal es el número de canal de la ventana correspondiente

La función devuelve un número (pos) que es la posición en memoria de la base del bloque de definición del canal correspondiente. A partir de ese dato pueden hallarse las características de una ventana dada con sólo "peekear" en el lugar correspondiente.

En la siguiente tabla, se resume la forma de hacerlo.

DATO			TIPO	POSICION RELATIVA
VENTANA	POSICION	X	W	24
		Y	W	26
	TAMAÑO	X	W	28
		Y	W	30
BORDE	ANCHO		W	32
CURSOR	POSICION	X	W	34
		Y	W	36
CURSOR	TAMAÑO	X	W	38
		Y	W	40
FUENTE	1 DIRECCION		L	42
	2		L	46
PANTALLA BASE			L	50
COLOR MASCARA	PAPEL		L	54
	STRIP		L	58
	TINTA		L	62
CARACTER	ATRIBUTOS		B	66(**)
CURSOR	FLAG		B	67(0=suprimido)
COLOR	PAPEL		B	68
	STRIP		B	69
	TINTA		B	70

```
BORDE     COLOR              8          71
ESTADO NUEVA LINEA           8          72 (0= implícito)
MODO      FILL               8          73 (0=off,1=on)
GRAFICOS ORIGEN Y            Float      74
GRAFICOS ORIGEN X            Float      80
GRAFICOS FACTOR ESCALA       Float      86
```

Para obtener un dato

```
dato = PEEK   (pos+POSICION RELATIVA); REMark si TIPO =8
dato = PEEK_W (pos+POSICION RELATIVA); REMark si TIPO =W
dato = PEEK_L (pos+POSICION RELATIVA); REMark si TIPO =L
dato = PEEK_L (pos+POSICION RELATIVA+2)*(2^(PEEK_W(pos + POSICION
RELATIVA)-2079)) : REMark si TIPO = Float
```

donde pos=WIN_VARS(#canal)

La función WIN_VARS fué publicada en QL WORLD del mes de Septiembre de 1987 en la sección OPD' CHANNEL, por Brian Huxley (Ilford, Essex). Allí puede encontrarse el listado en ensamblador

La función WIN_VARS devuelve un error "canal no abierto", si es el caso o "parámetro incorrecto" si el canal no del tipo de ventana. Es compatible con Q_liberator y Supercharge

El otro fichero suministrado (VENTANAS_BAS), es un programa en Superbasic que carga las extensiones e incorpora una función en Basic ventana(#canal)). Esta función en BASIC es un ejemplo del funcionamiento de la función residente WINDOW_VARS, que imprime en pantalla los valores actuales de algunos de los datos de la tabla anterior para el canal especificado.

NOTAS

(*) Dos palabras largas definen el origen de la dirección de la fuente de caracteres.

** Cada 'bit' de este "byte", define una característica de los caracteres, activada cuando el bit es 1, y desactivada cuando es 0.

```
Bit 0  subrayado          Bit 4: doble altura
Bit 1: flash              Bit 5: anchura extendida
Bit 2: transparente       Bit 6: doble anchura
Bit 3: XOR                Bit 7: caracteres posicionados en la escala
gráfica
```

Ernesto de Jesus
Madrid (Clave 56

COMENTARIO DE PROGRAMAS

EL COMANDO TK2 EXT DEL SUPERTOOLKIT (I)

En un número reciente de CLAVE, alguien preguntaba sobre la utilidad de la instrucción TK2 EXT del SuperToolKit I. Dada la oscura explicación que se da de dicha instrucción en el manual, creo que es interesante detenerse un poco en explicarla.

El QL y su Basic extensible: se le pueden añadir nuevas instrucciones (procedimientos o funciones). A estas nuevas instrucciones que no se encuentran en el QL original se les suele llamar extensiones. Cada vez que se conecta a red un disco, se 'recuerda' al QL, será necesario enseñarle las extensiones que debe reconocer. A este proceso se le llama inicialización de las extensiones. Como el proceso de inicialización de una extensión es muy común en el QL, existe una rutina en el QDOS que lo realiza (en concreto la rutina vectorial número $110, de nombre BP.INIT). Una extensión se compone de una 'palabra clave' (dicho de otra manera, el nombre propio con el que se la reconoce) y una 'definición' (es decir, el trozo de código máquina que define lo que tiene que hacer el ordenador cuando se teclea dicha 'palabra clave'). Para inicializar un grupo de extensiones, hay que llamar a la rutina de inicialización dándole como dato (en el registro A1) el lugar de la memoria donde se encuentra la lista que contiene las 'palabras clave' y la posiciones de las 'definiciones' de cada una de las extensiones a inicializar. Esta lista tiene la forma siguiente:

```
palabra     número de procedimientos
    Para cada procedimiento:
    palabra     posición de la definición (relativa a esta posición
    byte        longitud del tamaño del nombre
    bytes       caracteres del nombre
palabra     0
palabra     número de funciones
    Para cada función
    palabra     posición de la definición (relativa a esta posición
    byte        longitud del tamaño del nombre
    bytes       caracteres del nombre
palabra     0 para indicar el final de la lista
```

Un fichero de extensiones contiene básicamente el siguiente código:

; En la posición 0 -al principio del código-, las instrucciones necesarias para inicializar las extensiones, que en lenguaje de alto nivel serían:

```
            Pon "a" en A1
            Salta a la rutina de inicialización (BP.INIT)
            Vuelve al Basic
```

y en ensamblador sería una cosa como:

```
            ;
BP.INIT     EQU     $110
            ;
            MOVE.W  BP.INIT,A2
            LEA     PROC_DEF(PC),A1
```

```
        JSR     (A2)
        *
FIN
        RTS
        *
```

donde PROC DEF viene a ser lo que antes hemos llamado "x" (es decir, la etiqueta que marca la posición de la lista de definición de procedimientos y funciones

b) A continuación, en la posición "x", la lista de definición de los procedimientos y funciones

c) Finalmente, el código de cada una de las extensiones

Para cargar las extensiones contenidas en dicho fichero, es necesario reservar la memoria necesaria en el area de procedimientos residen es RESPR , cargar el fichero en dicha area de memoria RAM (LBYTES) y llamar (CALL) al codigo b tal como normalmente al principio del fichero, que inicializa los parametros y funciones cuya lista se da en b y cuyo código se da en c)

En el caso de extensiones en cartucho ROM, las operaciones anteriores son innecesarias ya que el "fichero" se encuentra permanentemente cargado en memoria ROM, en este caso y la inicialización de las extensiones la realiza automaticamente el QDOS durante la operacion de reinicialización del aparato. De esta forma, las extensiones se encuentran directamente disponibles tras pulsar "F1" o "F2"

De la forma descrita hasta ahora, los nombres de las extensiones son incorporados a la "tabla de nombres" del SuperBasic (la tabla con todos los nombres de variables matrices, procedimientos y funciones en SuperBasic o en codigo maquina ¿Pero que pasa si se han incorporado dos extensiones con el mismo nombre? En dicho caso, habra dos cosas, dos definiciones distintas que responden a la misma "palabra clave" ¿A cual de los dos codigos hara caso el QL?. La respuesta es que el último que haya sido inicializado.

La version reducida del Supertoolkit II, que viene en la ROM de los controladores de disco de QuJUMP, y la version completa del Supertoolkit II tienen una serie de comandos comunes (WCOPY, MREN, etc) Las definiciones de un mismo comando en uno y otro toolkit pueden tener algunas diferencias (algun error eliminado, un funcionamiento algo diferente) por ser de versiones diferentes. Así por ejemplo en el caso del controlador de disco DFLP, los comandos "wild card" (WCOPY etc hacen las preguntas 'Y/N/A/Q?' en el canal #1, mientras que en el Supertoolkit no hacen nada lógicamente en el #0 Para evitar esta superposición, QJUMP parece seguir la siguiente táctica

- Los controladores de disco son asignados al toolkit reducido sólo inicializan, al 'resetear' el aparato, las extensiones especificas al controlador de disco FLP_OPT FLP_USE, etc) y que, por tanto, no se superponen al toolkit completo Pero se da un comando (FLP_EXT) que, si se desea, inicializa las extensiones del toolkit reducido (por ejemplo, si no se tiene el completo) El Supertoolkit II completo, cuando se vende separadamente para conectar en la tarjeta de expansion de ROM, se inicializa automaticamente.

- es contra adores de disco con también el toolkit completo sólo s c a zan a reservar' aparato, las extensiones del reducido y es necesar o usar e comando 'TK2 EXT' para inicializar el completo

Como puede observarse, TK2_EXT y FLP EXT sirven para inicializar las ex ens ones de. toolkit correspondiente (el supertoolkit II completo y la versión reduc da de la ROM del contro.ador de disco, respectivamente), en los casos en los que és os no se encuentren d.sponibles automáticamente después del 'reseteo' ¿Que pasa s se emp ean d.chos comandos posteriormente?. Veamos los siguientes ejemplos.

```
Juego con controlador de disco QUME y Supertoolkit II independ ente
Inicio  Disponibles los comandos del Supertoolkit II.
        No disponibles los de toolkit reducido del controlador
FLP_EXT: Disponibles también los comandos del toolkit reducido
        En el caso de los comandos comunes al Supertoolkit II y al
        controlador de disco, es la definición de este último la que
        vale, al haber sido inicializados posteriormente.
TK2_EXT: Se reinicializa el SuperToolkit II. Por tanto, en el caso
         de comandos comunes, las definiciones válidas serán las del
supertoolkit.
etcétera,etcétera

Juego con SuperQBoard controlador de disco QUME y Supertoolkit II juntos).
Inicio: Disponibles los comandos del toolkit reducido del controlador.
        No disponibles los de Supertoolkit II.
TK2_EXT: Disponibles también los comandos del Supertoolkit II.
        En el caso de los comandos comunes al SuperToolkit II y al
        controlador de disco, es la definición del primero la que
        vale, al haber sido inicializados posteriormente.
etcétera,etcétera
```

Espero que de esta forma se haya comprendido la utilidad del comando TK2_EXT 'refuerza las definiciones del Toolkit II de los comandos y funciones comunes' de acuerdo al manual del Toolkit II).

Ernesto de Jesús Alcañiz
QUAVE-56 Estrasburgo

MAS SOBRE DIGITAL C & SUCCESS

Hoy es 13 de marzo y no sé si mi advertencia llegará a tiempo. El Digital C no es como puede ser tal. Precision, sino una copia casi exacta del small C con casi todas sus limitaciones. El autor la hizo en un primer momento para cederla a la Librería de Quanta. Está recomendado para principiantes del lenguaje C. Su limitación en programas ejecutables a 64K (incluyendo runtime dataspace), solamente usar números de 16 bits y los números en coma flotante almacenados en 3 bytes, en vez de las tres palabras que usa el QL normalmente. Y si encima agregamos la posibilidad casi absoluta de no poder insertar código en ASSEMBLER creo que tal como es esta primera versión del Digital C no debería haberse sacado nunca para un QL que usa un poderoso MC 68008.

El tiempo que se tarda en compilar el programa ejemplo SORT.C unos 6 archivos es de 45 segundos con el Paser más 15 segundos con el Code Generator (antes hecho a ojo con un reloj C TIZEN automatic de pulsera que adelanta cinco minutos por semana pero no gasta pilas ni tienes que darle cuerda).

El SUCCESS sobre la propaganda es el emulador de CP/M más potente que existe para el QL, pero en la práctica es peor de los cuatro que existen para el QL actualmente. La memoria está limitada a 64K. Esos 64K son razonables para un Z80, pero la mayoría de las maquinas CP/M tienen cientos de Kbytes de memoria segregada, lo que significa que los programas de la propaganda no pueden ser cargados en ese emulador. Posee otro emulador que se reproduce como conejos en el interior de QLave sin instrucciones, y su nombre es CP/MULATOR v 1.30 Peter Brymanski © 1987 (QLSOFT). Ese emulador es capaz de correr los programas ejemplo del SUCCESS, el Basic Microsoft Rev. 5.1 (el SUCCESS no puede correrlo) y muchos más. Me habría gustado haber comprado ese emulador, pero no sé ni de dónde ha salido ni su origen es un misterio que solamente conoce un socio de QLaver. Es más rápido y fácil de usar que el SUCCESS. Además usa el mismo formato que el QL y de un fichero disco como uso el SUCCESS. Además hay que agregar que la versión que me han enviado del SUCCESS tiene BUGS, pues muchas rosillas se niegan a funcionar.

Ay Ian Watson (autor del SUCCESS) me ha escrito contestándome mi carta e invitándome a que cambie mi copia por la última versión que tegan él, no tiene o no ha encontrado ninguno. También me ha comunicado que no sido el primero en quejarme. Esta primera versión corre el DBASE II y SUPERCALC de los AMSTRAD CPC y PCW.

Salvador Merino
Fuengirola. QLave-154

Programa: SpeedScreen
Editor : Creative CodeWorks
Precio : 20 libras (versión en disco y microdrive, sólo para el QL
          30 libras (versión en ROM para el QL o el THOR)
Versión: 1.26

Uno de los requerimientos a la hora de escribir la ROM del QL fue que las rutinas debían ser lo más compactas posibles para que cupieran en el 4K. La hora de escribir caracteres en pantalla, las condiciones tamaño, tipo de caracter, negrita, subrayados o no, se pueden ser muy variadas y, dependiendo de ellas, una rutina de escritura de caracteres en pantalla puede ser más rápida que otra o viceversa. A la hora de escribir en pantalla para el QL, se prefirieron las más generales es decir aquellas que podían escribir con un mismo código caracteres en las condiciones más diversas que las más rápidas pero que, al necesitar un código diferenciado para cada una de las condiciones, eran más largas. Este paquete sustituye esas rutinas por otras más rápidas (en promedio de 4 a 12 veces). Estas rutinas sólo sirven para el modo 4 y están empleadas las del QL para el modo 8 (por tanto sólo se gana en velocidad en el modo 4, salvo en el caso de CLS y SCROLL que también son acelerados en el modo 8). Las rutinas se cargan en memoria al salvo si se tiene la versión en ROM) después de realizar en QL con las sentencias habituales a=RESPR(longitud del fichero):LBYTES mdv1_speedscreen_code,a:CALL a

Cuando se recibe el paquete, lo primero que hay que hacer es leer el folleto de instrucciones de 12 páginas que, por supuesto, está en inglés (lo en Basman, si se prefiere). Para los que no dominen el inglés a continuación se detalla lo esencial.

El programa no se encuentra protegido, por lo que puede hacerse una copia por los procedimientos usuales. Sin embargo, cada copia de SpeedScreen tiene un número de serie y Creative CodeWorks asegura que los ficheros llevan ese número grabado en clave, de manera que puede averiguar inmediatamente de qué es el original a partir del cual se hizo una copia pirata. El original del programa contiene los siguientes ficheros

-- BOOT : Fichero de carga con sólo la sentencia EXEC mdv1_CONFIG_TASK
-- CONFIG_TASK: programa ejecutable principal
-- RAW : varios ficheros de datos, utilizados por CONFIG_TASK
-- QLJUDGE_TASK: un editor de fuentes de caracteres para el QL
-- _FONT: varios ficheros de fuentes alternativas a las standard del QL
-- PRINTSPEED_BAS: un programa de demostración en BASIC
-- WHATSUP_DOC: la documentación de las instrucciones del programa

El original del programa no es utilizable directamente sólo sirve para hacer una copia (de trabajo configurada) de acuerdo a los deseos del usuario.
La copia contendrá los mismos ficheros que el original a excepción de los tres primeros que serán sustituidos por

BOOT: RESPR,1BYTE 5 : CALL al fichero SPEEDSCREEN_CODE
- SPEEDSCREEN CODE: fichero que, cargado en el QL aceleraré la salida a pantalla

Para hacer la copia configurada, se debe colocar en Mdv1 el original o una copia con los mismos ficheros, conducta que, aunque prohibida por Creative Codeworks es más prudente para evitar daño al original , es preparar otro medio donde grabar la copia de trabajo, ejecutar el programa principal (SPEEDSCREEN_bas) mediante LRUN mdv1_BOOT , y seguir las instrucciones que aparecen en pantalla

Primeramente aparece la lista de las ocho posibles configuraciones diferentes del fichero 'speedscreen_code' junto con el tamaño (5 % Kbytes para la versión más sencilla 12 5 Kbytes para la más completa) y una somera descripción de cada una de ellas. Las versiones difieren en la incorporación o no de las siguientes rutinas

-- Turbotext: rutinas muy rápidas pero que sólo sirven para unos códigos determinados caracteres de la fuente estándard del QL o de aquella que se haya especificado al configurar SPEEDSCREEN - ver más abajo) no sobrevalgas, en cuyo caso 1 O y que comiencen a escribirse en una columna de 'pixels' multiplo de ocho; además sólo son usadas para imprimir, pero no para editar o para INPUT. Si no se incorporan estas rutinas o no son apropiadas, se utilizarán las de Colour Synthesis.

-- Colour Synthesis: rutinas menos rápidas pero que sirven para todas las condiciones salvo Caixa 2 o Mode 8, en cuyo caso se utilizan las rutinas del QL . Si no se incorporan estas rutinas, se utilizarán las del QL.

-- Fast Spaces: El caracter 32 -espacio en blanco- es uno de los más utilizados. Esta rutina lo imprime de una forma especial muy rápida. Algún programa puede redefinir este caracter por lo que se da la opción a incorporarla o no, ya que esta rutina imprimirá siempre el caracter 32 como un espacio en blanco

-- Fat Fonts Colour Synthesis: Aunque en las fuentes del QL, cada caracter puede ser de hasta ocho columnas de ancho, en las fuentes estándard sólo se encuentran definidas, más las rutinas de impresión sólo utilizan, las columnas 2 a 6 . Los caracteres de 6 columnas de ancho (caixa 0) se componen con estas seis columnas a las cuales se le añade una en blanco. Los caracteres del resto de anchos se logran poniendo más espacios en blanco y/o duplicando las columnas. Algunos programas de 'desktop' utilizan fuentes de caracteres de 8 columnas.
Esta versión especial de Colour Synthesis puede manejar dichas fuentes pero es 6K más larga y entre un 3 a un 7% más lenta.

Las versiones de Speedscreen son las siguientes:

p - Una versión reducida de Colour Synthesis, pero que puede ser usada con los programas de PSION o THE EDITOR en sistemas de 128K

```
        rbo ext con  ast Space
    s    olour Synthesis con Fast Space
    c    olour Synthesis sin Fast Space
    n   Turbotext y Colour Synthesis con Fast Space
    z = Turbotext y Colour Synthesis sin Fast Space
    4 = Turbotext y Fat Fount Colour Synthesis con Fast Space
    z = Turbotext y Fat Fount Colour Synthesis sin Fast Space
```

A continuación, el programa preguntará el medio en el que se desea realizar la copia (cinta o disco, el valor por defecto es el de la original). Si se ha elegido una opción con Turbotext, el programa preguntará la fuente a utilizar. Si se pulsa sólo ENTER, se utilizará la fuente estándard de la ROM, pueden, sin embargo, utilizarse otras (por ejemplo, una de las que vienen con el programa para lo que basta con entrar el número del fichero que se contiene. Como las rutinas de Turbotext sólo se emplean para imprimir, para INPUT o edit se emplearán las fuentes normales del QL.

Finalmente el programa realiza la copia, acabando con un reconocimiento de pantallas y con el código de carga (RESPR,LBYTES,CALL) del fichero creado. Dicho código es grabado también en en forma de fichero BOOT.

Se reinicializa el QL y se carga el fichero SpeedScreen. El QL funcionará normalmente para la pantalla, y los programas que, como QUILL, utilizan mucho tiempo la pantalla, irán mucho más rápidos. No sólo la salida de caracteres a pantalla es acelerada sino también el borrado de pantallas (CLS) y el SCROLL. La única diferencia observable es el funcionamiento más rápido del comando Mode e que sólo pone la pantalla en negro, pero no redibuja el papel de cada ventana activa. Además, salvo en el caso de la versión reducida 'n' se tendrán las siguientes nuevas extensiones:

- _SCROLL: cada vez que se "arrastra" la pantalla, se tarda mucho más tiempo en moverla que en escribir la línea. Con SPEEDSCREEN, en comandos como LIST, VIEW, COPY TO SCR_, la pantalla se mueve hacia arriba dos líneas a la vez en vez de una, con lo que se gana el doble de velocidad. Con el comando _SCROLL se puede modificar el número de líneas que se mueve la pantalla a la vez. _SCROLL 2 es el valor de omisión.

- _FOUNT (n fuente1,fuente2: comando similar a CHAR_USE en el SuperToolkit. Permite utilizar la nueva fuente de caracteres almacenada en fuente1: por ejemplo, fuente1=RESPR(875): LBYTES "mdv1_fantasy_fount",fuente1 en el canal #n. Si 'fuente1' o 'fuente2' son cero, se utilizará la fuente de la ROM del QL. Con el programa se suministran cuatro fuentes de caracteres de 875 bytes de longitud que, desgraciadamente, sólo incluyen los caracteres 32 a 127, y una fuente con los caracteres 127 a 191 que es igual a la segunda fuente de la ROM del QL. Cuatro de las fuentes (SER.F, FANTASY, SCIFI y GOLDEN) son de 8 columnas de ancho, por lo que deben ser impresas con las versiones FAT FOUNT de SpeedScreen e incorporadas a Turbotext.

Para [illegible] a modificar [illegible] fuentes [illegible] para [illegible] [illegible] [illegible] [illegible] editor de fuentes QUIDGE_TASK (EXEC_U modo) QUIDGE TASK)

XSTEP &canal,columnas e ySTEP &canal,filas son similares al comando HOR NO de SuperToolkit ! Permiten modificar la separación entre los caracteres de una forma más flexible que LSIZE

- SPEED 0 desconecta y SPEED 1 conecta las rutinas de SPEEDSCREEN

Para los usuarios de QRAM SPEEDSCREEN debe ser cargado antes que QRAM, o debe usarse SPEED 0; puede haber problemas en versiones antiguas de QRAM si se usan tanto en ambos modo 4 y en MODO 0 al mismo tiempo

Para los usuarios españoles: las versiones 1.21 y anteriores presentan hasta el 25 un error importante: los caracteres de la segunda fuente tienen las ventanas atenuadas [illegible] año, ahora eran impresos un poco (un pixel) más abajo de lo debido. Hace ante el envío del [illegible] Creative Codeworks facilitó una nueva versión para aquellos que observen este error.

En resumen un programa que todo usuario regular del QL debe tener a pesar de que el editor de caracteres no es ninguna maravilla.

Ernesto de Jesús Alcañiz
QUAVE-de Estrasburgo

COMENTARIO DE LIBROS

LIBRO :	El libro del RS 232
AUTOR	Joe Campbell
TITULO ORIGINAL	THE RS 232 SOLUTION
EDITORIAL	Anaya Multimedia
EDICION :	Segunda (revisada y ampliada
PVP :	2226 Pts.
PAGINAS :	712

Nos encontramos ante un libro bastante peculiar empezando por el tema del que trata. Realmente, como se dice en la misma obra existen pocas cosas en este nuestro cuadrilio de la "computación" tan desconocidas e incluso despreciadas y que tantos dolores de cabeza hayan causado, especialmente entre los recién llegados, como la conexión RS 232.

La lectura del libro convierte la visión que se tiene habitualmente de la conexión RS-232 como un mero enchufe en algo mucho más interesante, al menos desde el punto de vista del usuario que no se resigne a considerar su ordenador como una caja negra incognoscible sino que, muy al contrario, le guste conocer el funcionamiento de las cosas.

Centrándonos en el libro propiamente dicho, está dividido en dos partes claramente diferenciadas. La primera narra desde la historia del estandar RS-232 hasta la explicación de sus principios básicos pasando lógicamente por la descripción de los diferentes sistemas de comunicación entre ordenadores y periféricos, todo ello en un estilo ameno y claro. A nivel práctica, la primera parte del libro contiene un par de capitulos donde trata problemas reales de conexión entre equipos y como solucionarlos.

La segunda parte del libro es bastante diferente, pues trata de casos concretos de conexión de ordenadores entre si o con otros dispositivos. Realmente se trata de casos tan específicos que no serán de mucha utilidad a la mayoria de los lectores, aunque siempre se aprende algo. Por curiosidad, citaremos aquí los casos de conexión que se tratan en el libro, cada uno en un capitulo completo:

1. Ordenador SB-80 -- terminal ADDS
2. Ordenador NORTHSTAR -- impresora OKIDATA MICROLINE 83A
3. Ordenador SAYPRO -- impresora EPSON MX100
4. Ordenador OSBORNE I -- sintetizador de voz VOTRAX
5. Ordenador IBM PC -- impresora NEC SPINWRITER 3510
6. Ordenador APPLE MACINTOSH -- ordenador APPLE II C

No obstante, la segunda parte del libro incluye un par de capítulos adicionales sobre dos temas muy interesantes: uno sobre modems, donde se explica desde los principios elementales hasta cómo solucionar las conexiones más difíciles, y otro sobre herramientas de trabajo para comprobar y realizar conexiones (cables especiales, inversores, convertores, cajas de pruebas...)

En resumen, se trata de un libro muy instructivo y que a mi entender tiene varias aplicaciones:

En primer lugar, para el usuario de un ordenador que quiera conocer bien por encima y en profundidad los principios de las conexiones en serie entre dispositivos, en especial las conexiones RS-232.

En segundo lugar, para quien, teniendo ya algunos conocimientos sobre el RS-232, pretenda realizar alguna conexión particularmente compleja o bien profundizar en el tema. En el libro se dan datos y ejemplos suficientemente claros y detallados como para poder realizar cualquier conexión entre equipos RS-232 por extraña que sea.

Por último, decir que se trata de un libro totalmente recomendable para los interesados en el hardware en general. Y aunque en el caso de nuestro QL la conexión del ordenador a impresoras o a otros ordenadores con interface RS-232 no es nada difícil teniendo claros los principios básicos del estándar, siempre es interesante ampliar conocimientos.

Marcos Cruz (QLava-242)

LIBRO : QL PROGRAMACION EN SUPERBASIC
AUTOR : ROY ATHERTON
EDITORIAL : MICROTEXTOS
PAGINAS : 180
PVP : 1050 Ptas

Es un libro de introduccion a la programacion en Superbasic. No es como algun libro que he leido sobre el QL sólo una mera relación de comandos con sus funciones, sino que además de ideas sobre programación. Este es el punto que más me ha llamado la atención, no es que se extienda mucho en ese aspecto, pero da ideas fundamentales.

Creo que todos hemos oido decir que nuestro QL es ideal para emplear la llamada Programación Estructura, pero me gustaria saber cuantos son los que saben realmente lo que significa dicho concepto. A programadores experimentados seguro que les basta echar una ojeada a los nuevos comandos del Superbasic para coger esa nueva forma de programación, pero al resto, a los del montón, entre los que me incluyo, creo que no nos dice gran cosa. No me sorprenderia que incluso dichos programadores experimentados utilizaran los diagramas de flujo convencionales del Basic, en vez de utilizar los diagramas estructurales.

En el libro aparecen algunos programas comentados con sus correspondientes listados y diagramas estructurales y nos hace recorrer uno de ellos como si se tratara de un paseo. Dice el autor: "Imagine que es un insecto y que las lineas son paredes..." (la verdad es que dicho calificativo no denota mucho aprecio por nosotros, los novatos), bueno, bromas aparte, en el Capitulo 5 viene algo que considero interesante, se titula : ESTUDIO DE UN CASO SENCILLO y consta de los siguientes apartados: Idea, tarea o dificultad. Definición precisa del problema. Método. Planificación de datos o de pantalla. Análisis del problema. Diseño del programa. Programa. Salida. Comentario.

El libro consta de los siguientes Capitulos:
- El entorno del Superbasic
- Principios del Superbasic
- Control de la acción
- Manejo de datos
- Estudio de un caso sencillo
- Gráficos
- Datos
- Almacenamiento externo y recuperación de datos. Ficheros
- Operaciones y funciones
- Control de programas
- Puntos y ventanas

Además incluye 4 apéndices: Guía de referencia rápida de palabras clave. Mensajes de error. Sistemas de numeración. Angulos (éste último reconozco que es algo ridículo).

En fin, creo que un libro que puede sernos de utilidad a los principiantes, ya que al ser el autor el mismo que hizo la GUIA PARA EL PRINCIPIANTE que forma parte de la GUIA DEL USUARIO que se acompaña con el ordenador, pues quizás lime asperezas de algún socio que se hubiera atascado con el manual.

En el prólogo el autor dice que solo pretende conducir al principiante al punto donde es posible que empiece a pensar sensible y metodicamente sobre la resolución realista de problemas (copio literalmente) y anuncia un próximo libro sobre cuestiones mas "esotéricas", con programas de mayor longitud y profundidad, dicho libro aun no lo he encontrado.

Victor Librero
(OLave-184)

SUMARIO

1.- PORTADA.
2.- INFORMACION DEL CLUB.
3.- EDITORIAL.
4.- CORREO DE LOS SOCIOS.
20.- PREGUNTAS DE LOS SOCIOS.
23.- TRUCOS.
24.- COLABORACIONES DE LOS SOCIOS:

- 24 * CURSO DE FORTH (VII).
- 29 * COMANDOS GRAFICOS EN 3 DIMENSIONES (I).
- 31 * ORDENACION Y BUSQUEDA DE DATOS (II).
- 35 * PROCEDIMIENTOS DE ORDENACION.
- 50 * PROGRAMA VENTANAS.

52.- COMENTARIO DE PROGRAMAS:

- 52 * EL COMANDO TK2 EXT DEL TOOLKIT II.
- 55 * MAS SOBRE DIGITAL C& SUCCESS.
- 55 * SPEEDERSCREEN.

60.- COMENTARIO DE LIBROS:

- 60 * EL LIBRO DEL RS-232.
- 62 * QL PROGRAMACION EN SUPERBASIC.

64.- SUMARIO.

D.L.Z. 162 - 87